

ISaGRAF 自学培训手册

介绍和安装

ISaGRAF 概述

创建 ISaGRAF 项目

向 ISaGRAF 项目输出 SIXTAGS

使用快速梯形图编辑器

使用梯形图/功能块图编辑器

编译 ISaGRAF 应用

使用 I/O 仿真器

在 SIXNET Windows Run-time 方式运行应用

在 SIXTRAK 控制器中运行应用

使用在线调试器

用户自定义功能块

SIXNET China Tel: 010-6236290 Fax: 010-62316671 E-Mail: support@sixnetio.com.cn

本章我们将要：

- 安装 ISaGRAF 软件
- 安装 SIXNET ISaGRAF Windows Run-time软件
- 复制需要的培训支持文件

ISAGRAF 自我培训手册

本文档适用于熟悉SIXTRAK 硬件的用户。按步骤通过ISaGRAF IEC 1131-3编程，创建、编译、调试仿真、及运行一个ISaGRAF应用。

本教程为模块结构，每章是一个有针对性的单独的文档。本教程有一定的连续性，所以我们推荐（但不要求）您完成本手册的每一章。

本手册适用的项目文件和 **ISaGRAF Windows Runtime for SIXNET Control Room**一起提供，自动安装在正确的目录下。

TRAINING.6PJ 即 **SIXTRAK Plant Floor** 项目文件
该项目包括一个 ST-GT-ETH-24P 控制器和 Virtual DI 8, Virtual DO 8, Virtual AI 8, 及 Virtual AO 8 模块。本教程的运行不需要任何实际的硬件。如果您有不同于ST-GT-ETH-24P的其他SIXTRAK控制器，应当先对该项目文件作相应修改。

TRAIN_SX.PIA 即 **ISaGRAF** 培训项目文件
本培训手册将一步一步地带领您创建一个完整的 ISaGRAF 项目。如果您想预先看到完整的项目，可以用文档工具的恢复功能安装该文件到 ISaWIN目录。

SXON_OFF.AIA 即 **ON_OFF** 功能块
本培训手册将一步一步地带领您创建一个用户定义的功能块。如果您想预先看到完整的项目，可以用文档工具的恢复功能安装该文件到 ISaWIN目录。

ISAGRAF 安装

ISaGRAF安装程序在Windows程序组加上 "ISaGRAF"程序组，并在\ISAWIN\EXE子目录下产生一个初始化文件"ISA.ini"。按下列步骤安装ISaGRAF：

首先 → 插入 SIXNET 软件光盘，光盘将自动启动。

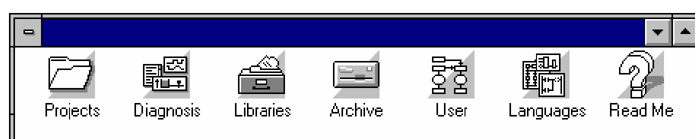
然后 → 从主菜单选择安装

然后 → 选择ISaGRAF Workbench和下列在线指南完全安装：

安装程序会询问是否安装下列组件：

- ISaGRAF可执行程序
- 在线信息和帮助文件
- ISaGRAF标准库
- ISaGRAF 样例应用

推荐在首次安装时安装所有选项，个别组件也可后来重新安装。当所有的 ISaGRAF 文件复制后，ISaGRAF程序组声称，ISaGRAF主目录为"\ISAWIN"。子目录也自动建立。



安装SIXNET ISAGRAF WINDOWS RUN-TIME软件（ISARUN）

安装ISaGRAF Workbench后，Isarun自动安装。

注意：　为了与SIXNET I/O正确地通讯，必须安装SIXNET ISaGRAF Windows Run-time。Isarun必须在Workbench安装后安装，重新安装ISaGRAF Workbench后，Isarun也应重新安装。

使用 ISAGRAF 在线帮助

在线帮助随ISaGRAF Workbench一起安装，包括以下主题：

- ISaGRAF语言参考手册
- 完整的用户指南 (对任何ISaGRAF工具)
- 标准库中元素的技术要点

使用 SIXNET 在线帮助

在线帮助随SIXNET ISaGRAF Windows Run-time一起安装。

本章我们将要：

- 概述ISaGRAF，及其结构和资源
- 定义基本的ISaGRAF术语
- 简述SIXNET ISaGRAF系统的设置步骤



关于ISaGRAF的详细信息请参考
ISaGRAF 用户指南

本章开始前：

您应当已经阅读了本手册的介绍，并对控制程序有基本的了解。

什么是 ISAGRAF

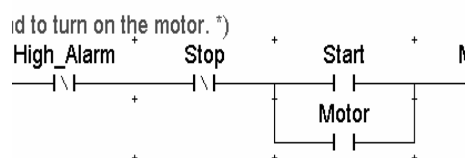
ISaGRAF 是容易理解和使用的控制系统编程环境，它使得SIXTRAK I/O和 VersaTRAK RTUs 成为高性能、低价格的控制器。ISaGRAF使用标准的工业PLC编程方法，不需要高级计算机语言或计算机硬件专业知识就可设计出高性能的应用。

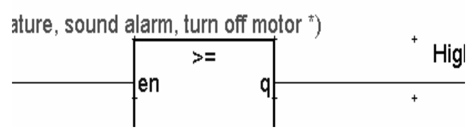
ISAGRAF 项目和程序

一个ISaGRAF项目**(project)**是一组程序和函数的集合，构成一个完整的控制应用。每个程序控制应用的一个特定部分。程序和函数在项目中按照其在ISaGRAF目标循环的位置分为四个部分。在同一个应用中可以混合使用所有五种IEC 1131-3语言。

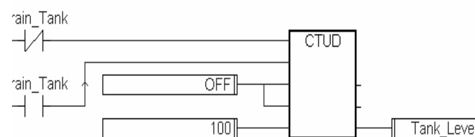
IEC 1131-3 语言

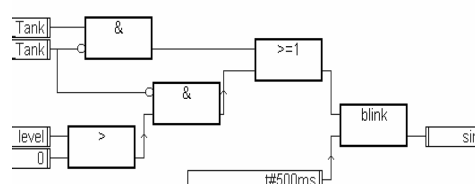
梯形图(LD) 是大多PLC支持的程序逻辑的类继电器触点和线圈的经典表示方法。梯形图可在**Quick Ladder**编辑器中快速容易地建立，集合基本元素而形成一个完整的程序。

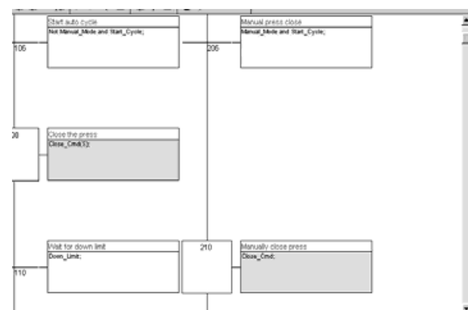




功能块图(FBD) 是高端过程控制器与DCS系统的通用编程语言，把图形化的处理计算功能加入了ISaGRAF。它可与梯形图元素一起使用以集合过程控制与机器控制的功能。







顺序功能图(SFC) 是强大的工业流程图语言，将项目组织为有顺序的(按步执行)操作。它使用简单的图形表示不同的步(Step)，并将这些步按布尔条件(变迁-Transition)链接起来，每步中的作用(action)由其他四种语言详细描述。

结构文本(ST) 是类Pascal的计算机编程语言，它对于编写复杂的操作很有效，常用于创建用户定义的功能块和描述SFC的步及变迁。

指令列表(IL) 类似汇编语言，提供给熟悉Siemens陈述列表编程的用户。

```
(* extract left and right parts of the string *)

right_part := right ( animation , 1 ) ;
left_part := left ( animation , 9 ) ;

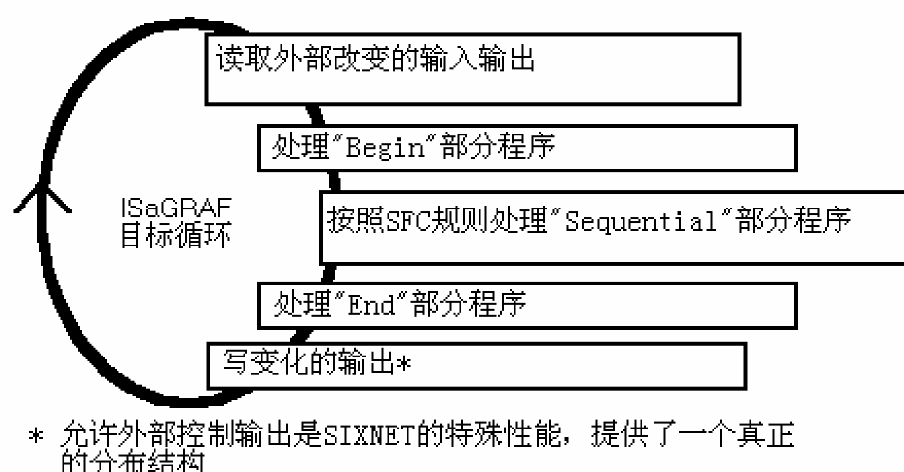
(* invert left and right parts to modify string *)

animation := ( right_part + left_part ) ;
```

Keywords	
:=	TRUE
FALSE	AND
OR	XOR
RETURN;	IF
THEN	ELSE
ELSIF	END_IF;
CASE	END_CASE;

ISAGRAF 目标循环：

SIXNET ISaGRAF runtime 按照如下目标循环执行一个控制程序：



ISAGRAF编程部分

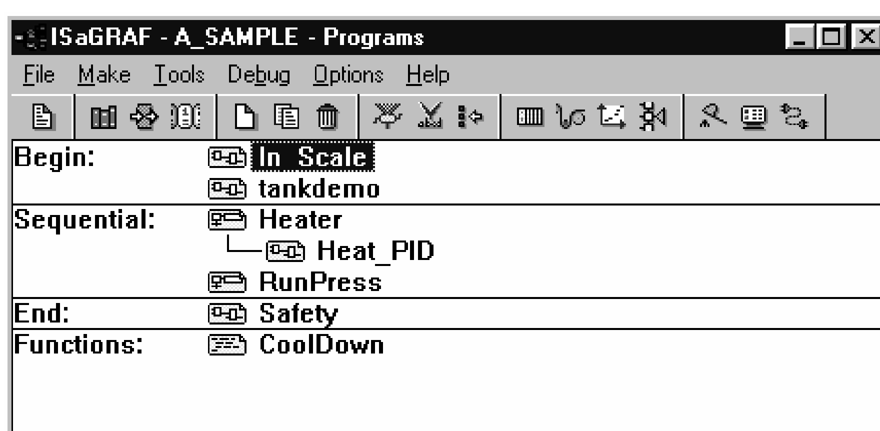
如前所述，每个ISaGRAF控制项目由一系列程序、子程序和函数组成，按照上图的目标循环分作四个不同部分。每部分可包含几个程序，也可根本没有。在ISaGRAF程序管理窗口，每部分有水平线条分隔。

Beginning: 循环的程序不依赖于时间。本部分程序在输入扫描后的开始部分由系统执行。不能是SFC。

Sequential: 遵守SFC规则和实现，依赖于一系列步和变迁。Sequential部分的程序支持程序间的平行处理编程和父子关系，必须是 SFC。

End: 与beginning部分程序有相同特点，在更新输出前的循环结束时执行。

Functions: 能够被其他三部分程序调用的子程序。函数不能是SFC。



ISAGRAF字典

本手册常常引用ISaGRAF字典，该字典是内部变量、输入变量、输出变量以及定义的集合，在项目程序中应用。

ISaGRAF - A_SAMPLE - Global booleans

File Edit Search Options Help

Name	Attrib.	Addr.	Comment
Close_Cmd	[internal]	0000	Command to close the press
Heat_Request	[internal]	0000	Command to apply power to the heater
sim_mixer	[internal]	0000	Used by the simulator to turn the mixer
OFF	[internal]	0000	Permanently False Variable
Start_Cycle	[input]	0000	DI0:DI:1 (X0) Auto Cycle Start
Down_Limit	[input]	0000	DI1:DI:2 (X1) Press Fully Closed
Up_Limit	[input]	0000	DI2:DI:3 (X2) Press Fully Opened
Safety_Gate	[input]	0000	DI3:DI:4 (X3) Safety Gate Closed
Manual_Mode	[input]	0000	DI4:DI:5 (X4) On for Manual Mode
Fill_Tank	[input]	0000	DI5:DI:6 (X5) On if Material is Needed
Drain_Tank	[input]	0000	DI6:DI:7 (X6)
DI8	[input]	0000	DI7:DI:8 (X7) also DO:8 (Y8)
Press_Valve	[output]	0000	DO0:DO:1 (Y0) On to Close Press
Main_Power	[output]	0000	DO1:

输入和输出变量是I/O值。可以从SIXTAGS输出，以节省建立输入输出变量的时间。SIXTAGS 输出在ISaGRAF字典中自动定义和组态I/O变量。

内部变量是用于计算和分析的逻辑名，既不是输入变量也不是输出变量。例如，可以建立一个量程转换的温度变量，以使用工程单位表示原始的I/O值。

定义是一个常数的说明性名字，以便容易地维护项目。由于一些说明性名字在整个项目中使用，如果需要改变数值，只需在字典中改变一次即可。

定义和变量可以在字典中创建，分作：

- **Local**（局部的），只用于一个程序
- **Global**（全局的），可用于单个项目的任何程序
- **Common**（通用的），可用于任何ISaGRAF项目

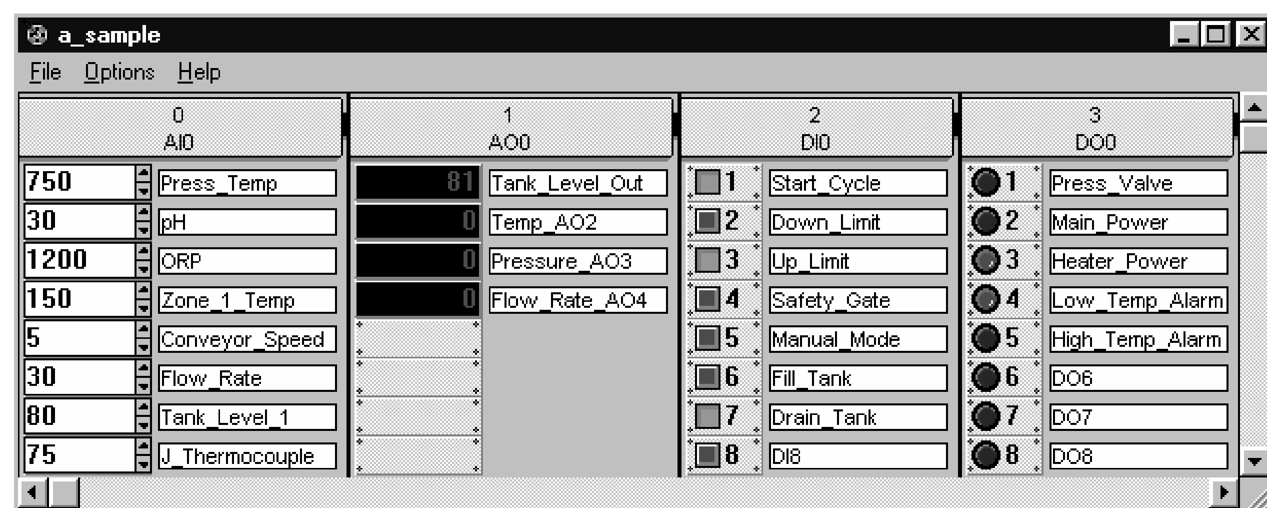
ISaGRAF字典的详细信息请参考ISaGRAF 用户指南

使用在线调试工具



所有的ISaGRAF编辑器具有 **"Debug"**（调试）命令，并提供一组观察目标站活动I/O数据的工具。 每个I/O 点的状态和数值在程序逻辑中图示出来，并允许用户写入I/O值以测试应用。本手册后面的章节将详细学习调试器（Debugger）。

使用I/O仿真器

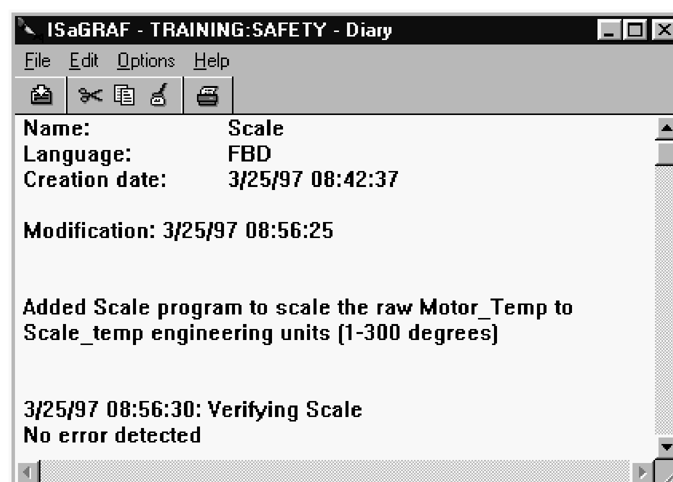


"Simulate"（仿真）命令可以仿真方式打开调试器（debugger）。仿真方式下，I/O仿真器窗口代替了链接外部I/O，使我们可以在连接到活动系统前测试程序。这个方便的开发工具节省了我们的时间，并减少了开始的故障。

程序日记

ISaGRAF包括一个与编辑的程序相联系的日记文件，编辑或修改程序时可以进行记录。可以由"**File / Diary**"命令手动输入。如果在程序编辑器的 "**Options**"菜单选择了"**Update diary**"方式，则每次程序保存前更新日记对话框会自动打开。

每次程序编译时，语法检查信息会自动记录到日记文件，并加上日期/时间标志。



使用文档特性（ARCHIVING）-- TRAINING项目

利用ISaGRAF文档(archive)工具可以保存ISaGRAF项目和重要文档或备份目录。

可将一个ISaGRAF项目所有的源代码、库元素和通用数据归档为一个完整的部分。文档工具的主窗口的**Options**，**Compression** （压缩）特性可用于减少备份文件的大小。ISaGRAF库（用户定义功能块，技术要点等）另外分别归档。

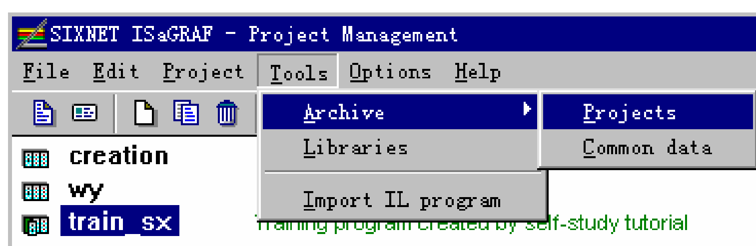
当你归档一个ISaGRAF对象时，文档管理器为对象分配一个特殊的扩展名，例如项目文件为“*.**PIA**敚　滠块蝊 “*.**AIA**敗Ｎ募　┳姑　耐暾　斜聿渭韄*SaGRAF用户指南* 的攅褂梦牡倒ぞ邷章节。

特别注意：

本手册相关的两个归档对象为：TRNGBKUP.**PIA** 和 SXON_OFF.**AIA**。TRNGBKUP是完整的ISaGRAF培训项目(training)，SXON_OFF是完整的ON_OFF功能块。可使用Archive, Restore（恢复）功能安装到ISaWIN目录下。

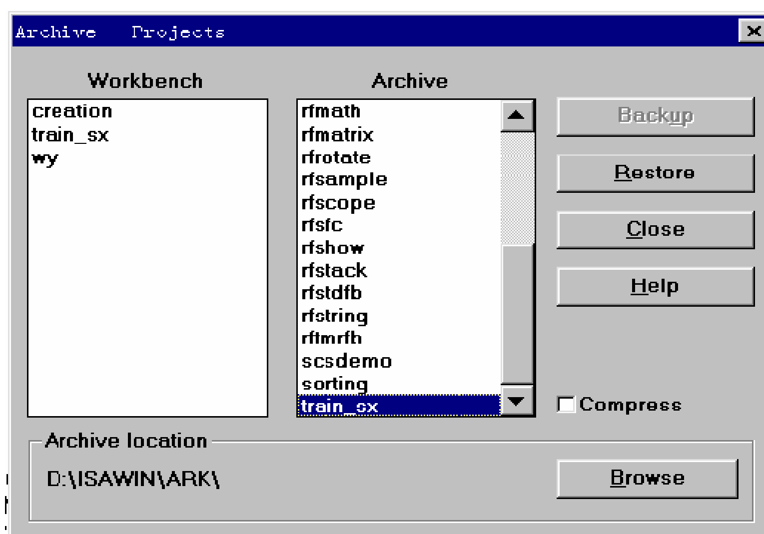
首先 →

打开ISaGRAF项目管理器，选择Tools/Archive， 选择Projects。



Archive主窗口左边Workbench栏是当前APL目录下的项目文件，右边Archive栏是ARK目录下的已归档文

件。



然后 →

从Archive栏选择train_sx，按Restore按钮，将其恢复到APL下。 关闭之。

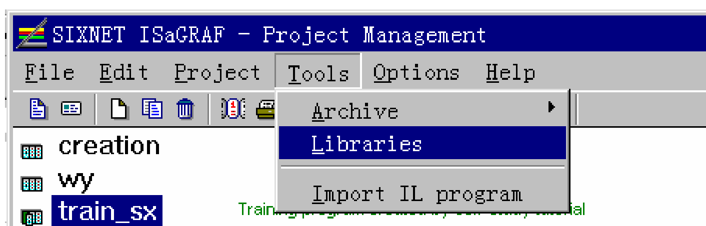
也可将当前项目文件备份（Backup按钮）到要求目录下。

可按下Browse按钮，从弹出框中选取文件和目录。

归档功能块：

然后 →

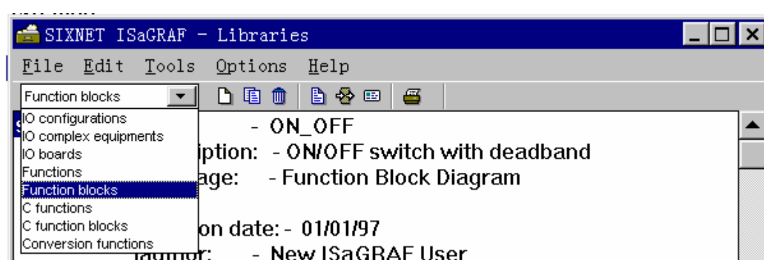
打开ISaGRAF项目管理器，选择Tools/Libraries。



然后 →

从弹出框Libraries的下拉框选择Function Blocks。

从菜单栏选择Tools/Archive。



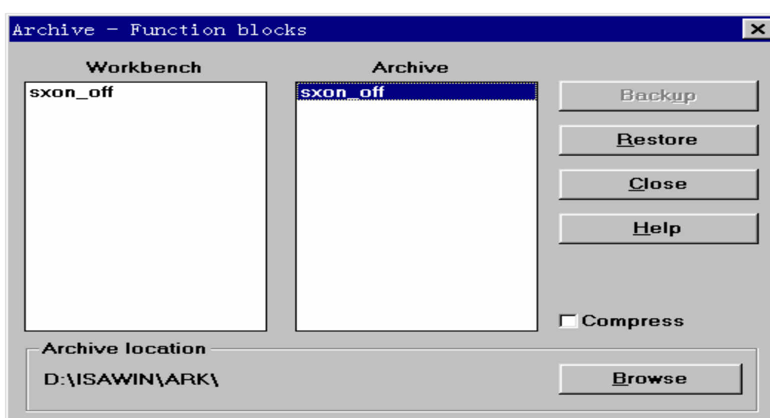
然后 →

弹出框Function Blocks的左栏为以归档的功能块，右栏为目前可用的功能块。

从左栏选取sx0n_off功能块，按下Restore按钮，即可将已归档功能块恢复出来。

也可将当前功能块备份（Backup按钮）到要求目录下。

可按下Browse按钮，从弹出框中选取文件和目录。



ISAGRAF 作为一个SIXNET项目的一部分

SIXNET提供了包含ISaGRAF Workbench在内的一个相互协作的Windows软件工具，这些工具包括Plant Floor，Control Room，SixTags等，以共享组态数据、简化工作。下面是创建实现SIXNET共享资源数据库的系统的步骤：使用ISaGRAF调试工具测试并下装程序。

步骤 1: I/O硬件组态

使用Plant Floor 软件对I/O站组态，给每个I/O点分配位号名，并设置系统特性。Plant Floor 包括test I/O等工具，以便启动后查找故障。

工具: SIXNET Plant Floor (ST-SPF)

步骤 2: 创建Windows共享资源数据库

如果要将I/O站链接到基于Windows 的应用程序（包括在计算机上运行的ISaGRAF），运行Control Room IOmap 以创建共享资源数据库 (DLL)；如果从Plant Floor启动IOmap，则项目将被打开，IOmap数据库自动建立。

工具: SIXNET Control Room (ST-SCR)

步骤 3: 将I/O位号定义输出给ISaGRAF项目

利用SixTags，将Plant Floor I/O位号定义输出给ISaGRAF。参考本手册输出位号章节。

工具: Sixtags tag dictionary utility

步骤 4: 编写ISaGRAF 项目

按本手册说明创建您的 ISaGRAF 项目文件。

工具： ISaGRAF Workbench

步骤 5: 仿真调试该项目

使用ISaGRAF I/O 仿真器测试项目。参考本手册揁褂肐/O仿真器所陆凇_

工具： ISaGRAF Workbench

步骤 6: 使您的系统在线运行

使用调试工具下载并测试程序。也可使用SIXNET Control Room 和 Plant Floor软件的强有力诊断工具。

工具： ISaGRAF Workbench 和 SIXNET Control Room 及 Plant Floor

本章您将要：

- 回顾ISaGRAF项目管理器元素
- 学习归档功能
- 创建ISaGRAF培训项目(Training)



关于ISaGRAF项目管理器的详细信息，参考ISaGRAF 用户指南

本章开始前，您应当：

完成了本手册的 *安装和介绍* 和 ***ISaGRAF概述*** 章。

ISAGRAF项目管理窗口：

一个ISaGRAF项目是一组程序(程序, 子程序, 函数, 等等)的集合，用于控制一个过程。一个项目对应于在一个目标控制器中运行的一个完整的过程。缺省情况下，ISaGRAF项目存储在:\ISAWIN\APL 目录下。

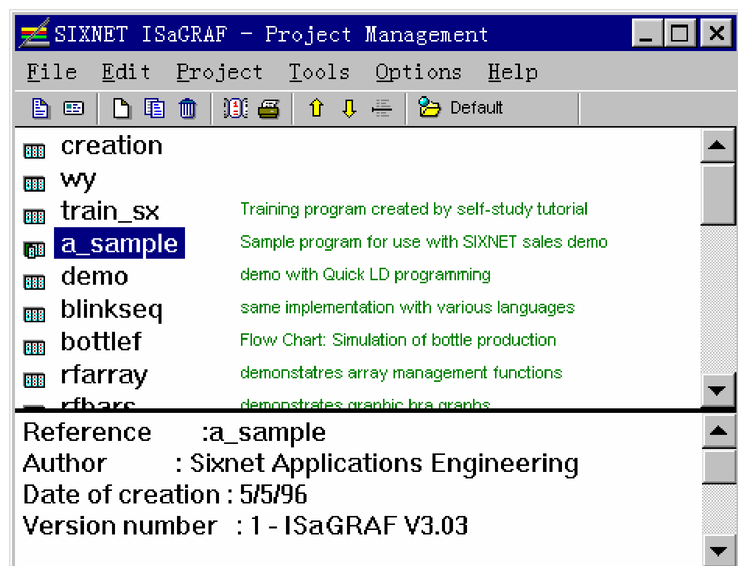
ISaGRAF项目管理窗口布局与说明如下：

ISaGRAF 项目列表

项目说明

创建项目时，必须遵守下列命名规则:

- *名字不超过8个字符*
- *首字符必须是字母*
- *其余字符可以时字母、数字或下划线*
- *项目名字不区分大小写*



ISaGRAF项目管理窗口分作两部分，上面部分是项目列表，下面部分是项目说明。项目说明描述项目创建的日期和时间，作者名，内容和目的等。

在项目管理窗口可以：打印项目，保存修改历史，归档和恢复项目，编辑项目说明等等。参见ISaGRAF用户手册的相关章节。

本手册创建的项目：

本手册将帮助您建立一个简单的应用，使用ISaGRAF Workbench监视马达的运行。

相应的SIXTRAK Plant Floor 文件：**TRAINING.6pj**，包含创建ISaGRAF项目所需的所有的I/O组态。使用的I/O有：

Start:	DI0	启动马达
Stop:	DI2	停止马达
Safety_Gate:	DI3	防止马达在危险条件下运行的安全门
Motor:	DO0	马达状态
High_Alarm:	DO1	马达温度过高报警
Motor_Temp:	AI0	马达温度

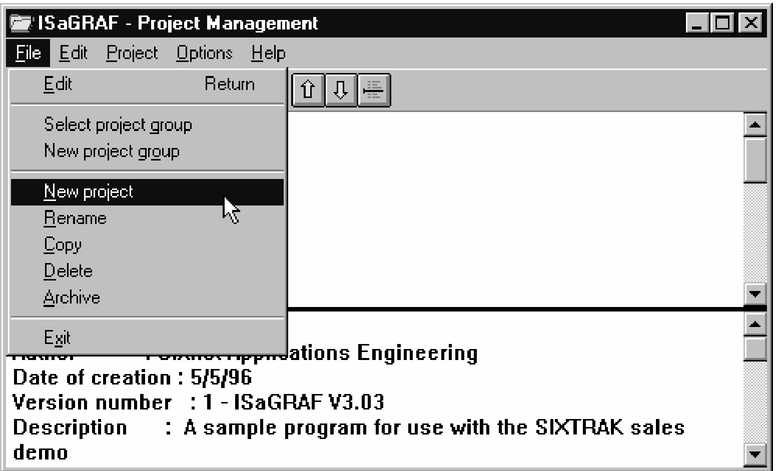
TRAINING.6pj也可能包含本手册步需要的位号。这是为您进一步开发应用所需准备的。

开始：

双击 图标打开ISaGRAF项目管理器。

然后 →

选择文件菜单（File），选新建项目命令（New Project）。

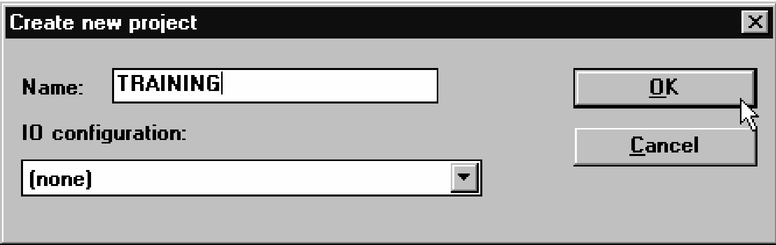


然后 →

将程序名定义为 **TRAINING**.

然后 →

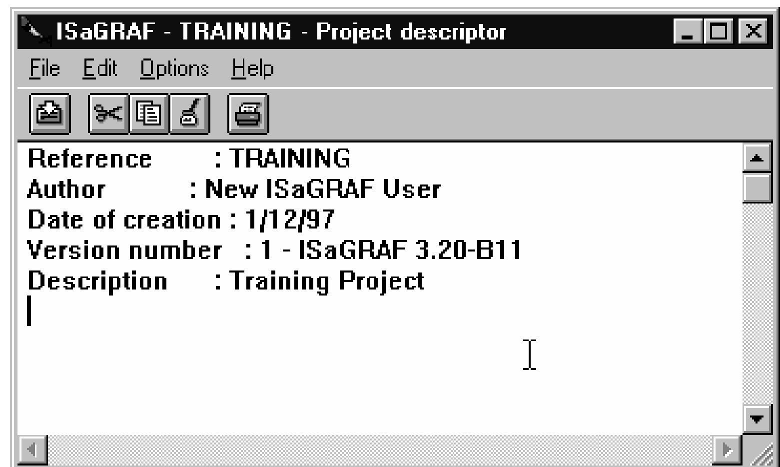
点击 **OK**.



然后 →

点击 项目说明图标（Project Descriptor），在对话框中输入你的信息。

选择 **File, Save** 及 **Exit** （保存，然后退出）。



然后 →

将项目管理窗口最小化。

本章我们将要：

- 使用从SIXNET Sixtags 工具，从Training.6pj项目将标签输出给ISaGRAF项目
- 确认标签限制
- 重新打开ISaGRAF TRAINING项目，确认输出成功



关于Sixtags的详细信息，请参考在线帮助

本章开始前，您应当：

完成了**创建ISaGRAF**项目章，ISaGRAF TRAINING项目已被建立。对SIXTRAK硬件及其组态工具应有所了解。

SIXTAGS工具 -- 一个共享标签数据库：

在SIXNET Plant Floor组态文件(.6pj)中，创建了模块及I/O标签名，使用Sixtags应用工具的输出功能，标签名可方便地输出到其他的Windows应用，如*Citect, Intellution FIX,* .CSV格式，当然也包括ISaGRAF。

SIXNET I/O Tag Dictionary - TRAINING.6PJ

File Edit View Define Help

	SIXNET I/O Tag Name	Tag Description
1	Motor_Temp	Motor temperature in degrees
2	Motor	Motor status
3	High_Alarm	Alarm if motor temperature reaches 200
4	Start	Start motor
5	Stop	Stop motor
6	Safety	Disables motor to run if unsafe
7		

For Help, press F1

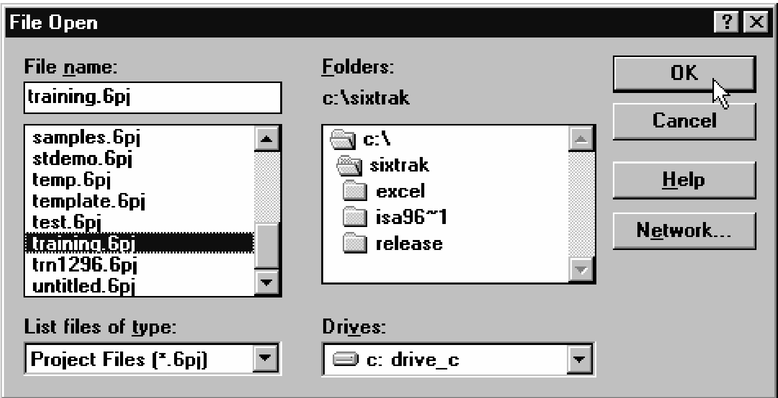
记录在ISaGRAF输出文件的标签包含Plant Floor中组态的I/O标签名和模拟I/O量程值。利用输出功能，可减少或取消在不同应用程序中建立和维护标签名的要求。

开始：

从Sixtrak程序组中启动SIXNET Sixtags工具。

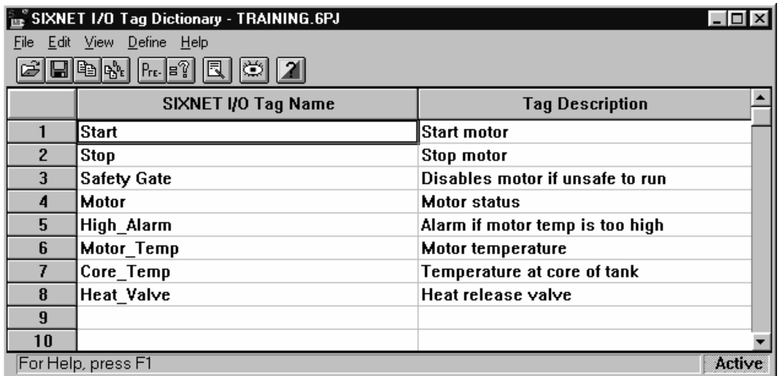
然后 →

选择File，Open，Project，然后选TRAINING.6pj。



然后 →

选择Define，Tags 和Links。



然后 →

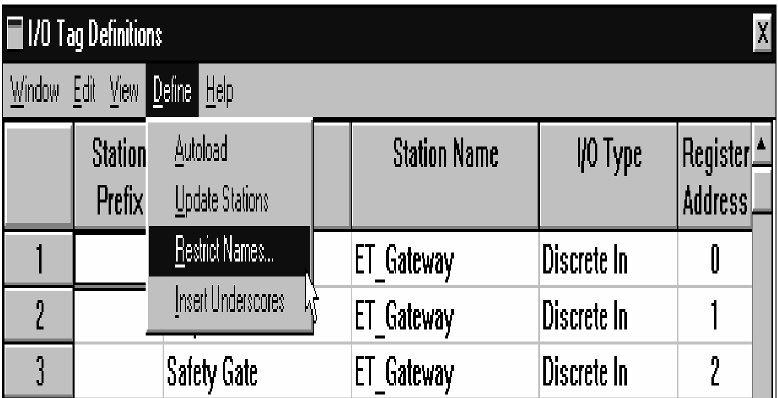
选择Define，Restrict Names。

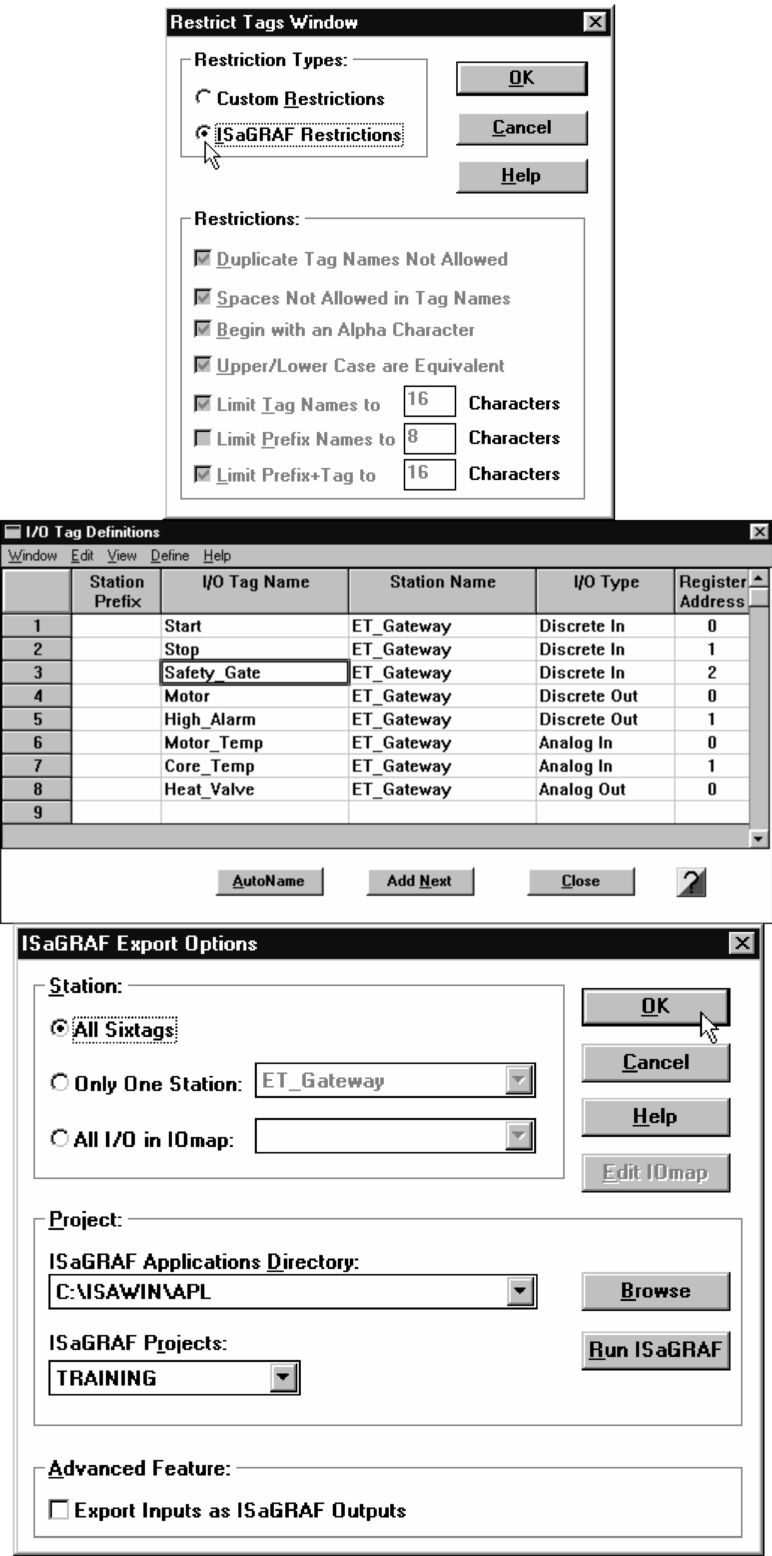
然后 →

选择ISaGRAF Restriction，点击OK

这个特性是确认所选择标签名是否符合ISaGRAF (IEC 1131-3)标准。

这时，Safety Gate标签变为黄色，表示它不符合标签名限制。





然后 →

双击**Safety Gate** 标签，将它改为**Safety_Gate**，关掉该窗口。

然后 →

选择 **File**, **Save** 保存。

然后 →

选择File，Export，ISaGRAF。

在弹出的对话框中，选择要输出的标签、项目所在的目录、项目名（**TRAINING**）。如右图。

点击 **OK**。

ISaGRAF输出窗口允许选择所要输出的I/O标签，可选择project, station, 或 Iomap。详细信息请参考SIXTRAK在线帮助。

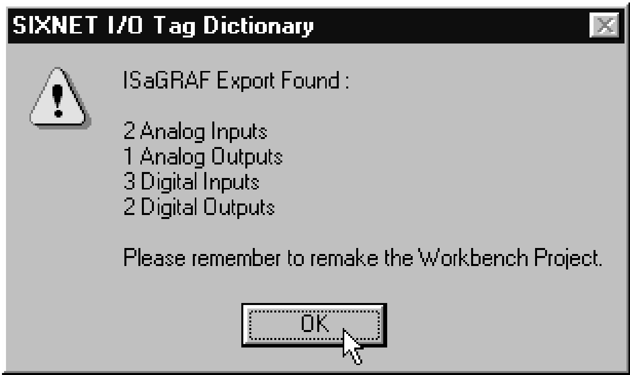
这时，Sixtags确认所输出的标签数。

然后 →

点击 **OK**。

然后 →

退出（**Exit**）Sixtags。

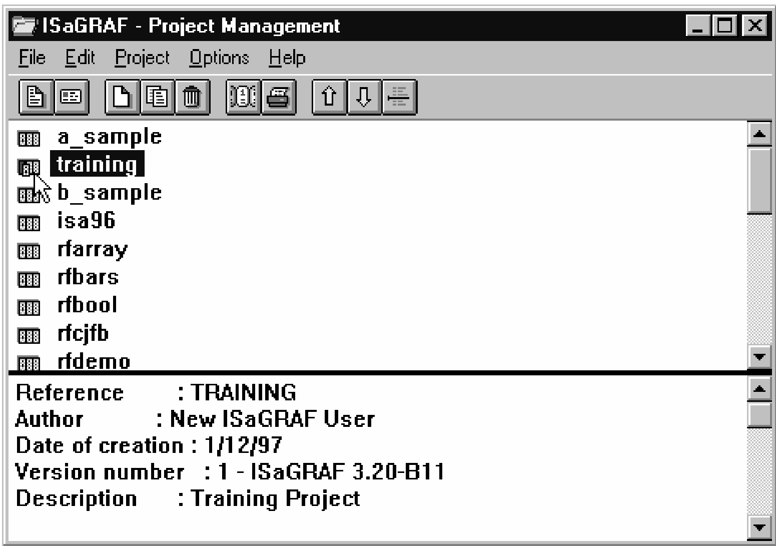


然后 →

最大化ISaGRAF项目管理窗口。

然后 →

双击项目名（training）图标，打开程序管理窗口。

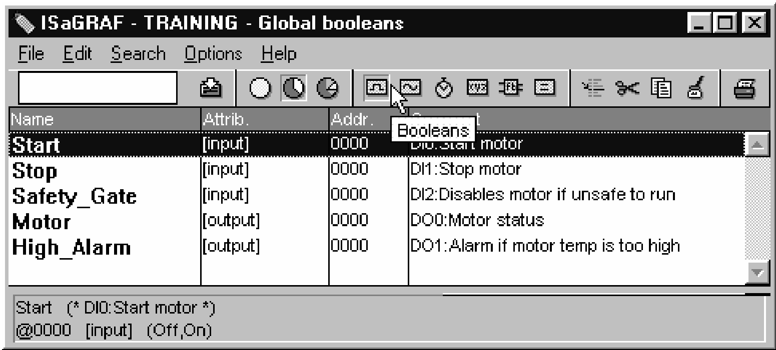


然后 →

选择**File, Dictionary**。

然后 →

使用工具栏上按钮，显示布尔I/O变量 (离散).



然后 →

使用工具栏上按钮，显示整形/实型I/O变量 (模拟).

字典应包含所有输出的标签。

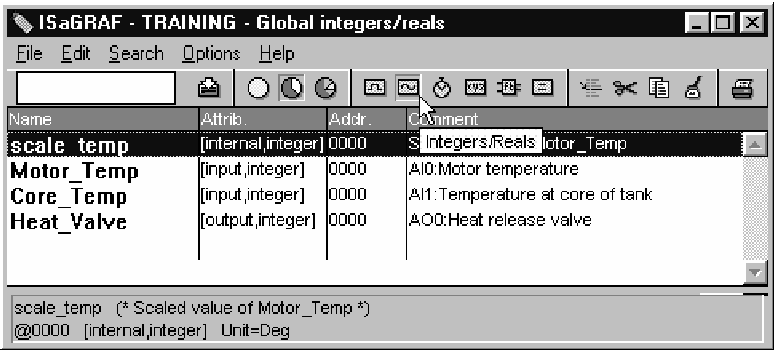
然后 →

关闭 **Dictionary.**

然后 →

关闭Training项目。

现在，您已为建立ISaGRAF应用做好了准备。



本章我们将要：

- 学习Quick Ladder（快速梯形图）工具
- 创建一个简单的梯形图程序
- 建立一个内部变量 Scale_temp
- 保存并确认这个程序
- 向这个程序将入一个功能块安全特性

梯形图编辑器的详细信息请参考ISaGRAF用户手册的相关章节。

开始本章前，您应当：

已经完成了本手册的*向ISaGRAF项目输出位号*章节，并且有较好的梯形图逻辑编程基本概念。

梯形图编程基础：

一个梯形图程序表示为由接点（离散输入）和线圈（离散开关输出）组成的一个阶梯列表。下面是梯形图（LD）的基本元件：

阶梯头 (左栏杆)

每个阶梯从左栏杆开始，它表示撜鏀状态。阶梯可有一个逻辑名（Label），用于跳转指示。当放置一个接点时，左栏杆自动产生。

接点

接点是按一个布尔变量状态而修改的布尔数据流，变量名显示在接点符号上面。ISaGRAF 支持下面的接点类型：

-| |-.................. 直接接点（常开接点）
-|\|-.................. 否定接点（常闭接点）
-|P|-.................. 上升沿触发接点
-|N|-.................. 下降沿触发接点

-( )- 线圈

线圈表示一个动作，由其左边的链接状态决定。符号上显示出变量名。ISaGRAF支持下面的线圈类型：

-( )-.................. 直接线圈
-(\)-.................. 否定线圈
-(S)-.................. 置位线圈
-(R)-.................. 复位线圈
-(P)-.................. 上升沿检测线圈
-(N)-.................. 下降沿检测线圈

功能块
在一个 LD 程序中，一个块表示一个函数、一个功能块、一个子程序、或一个操作符。其第一个输入和输出参数（开关量）总是连接到阶梯上。

阶梯尾（右栏杆）
一个阶梯的结尾，当放置一个线圈时，右栏杆自动产生。

跳转符号
跳转符号指向一个具有标号的目标阶梯，位于一个阶梯的末端，当该阶梯为真时，执行跳转。

返回符号
返回符号位于一个阶梯的末端，当该阶梯为真时，表示该程序必须停止。（扫描结束条件）

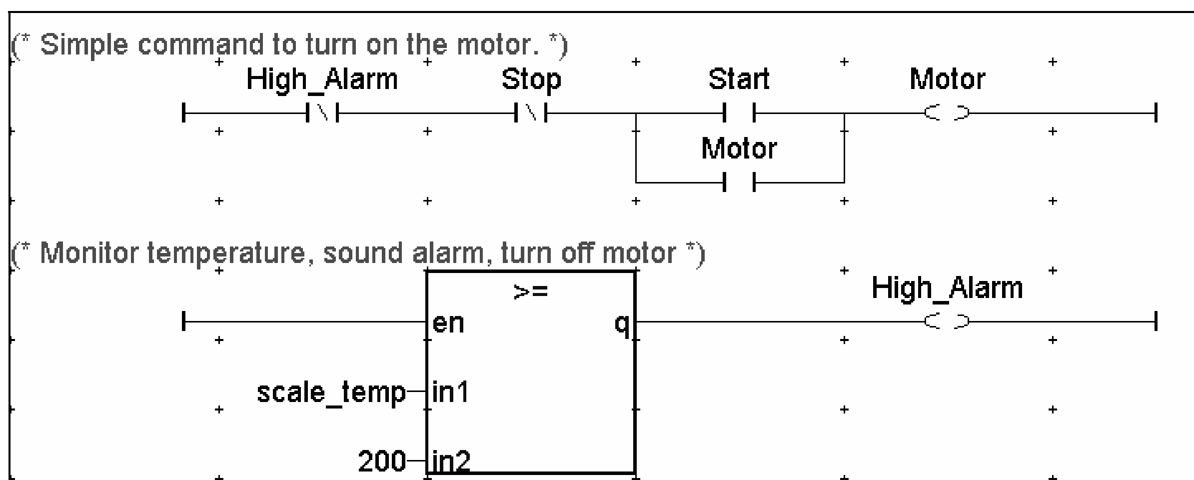
快速梯形图编辑器限制:
ISaGRAF快速梯形图编辑器不允许在线圈的右边插入接点或线圈。如果几个输出由同一个阶梯控制，相关的线圈应平行画出。



详细信息请参考快速梯形图编辑器在线帮助。

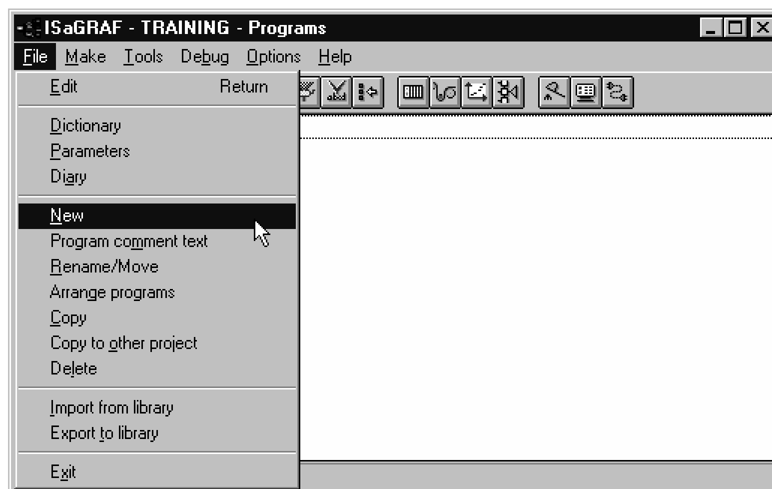
梯形图编程实例

本例中，我们将建立一个基本的梯形图：



开始：

从ISaGRAF项目管理器窗口中，打开TRAINING项目。（双击图标  ）



然后 →

从Training编程窗口，选择 **File, New**.

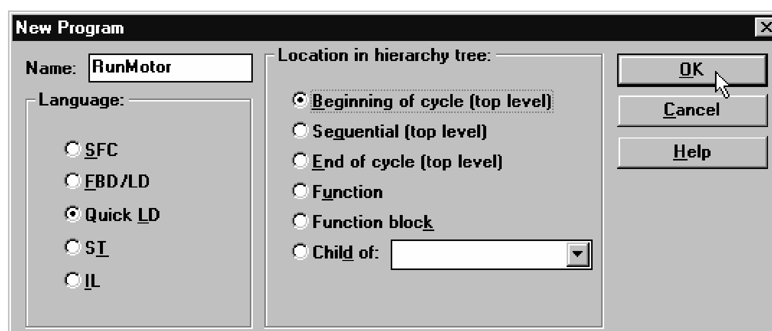
然后 →

填写程序名： **RunMotor**，

选择 **Quick LD** 作为编程语言，

选择 **Beginning of cycle** 作为其在层次树中的位置。

点击 **OK**。

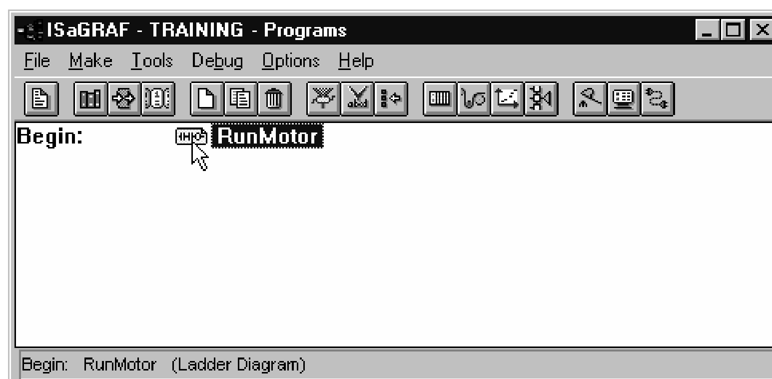




关于程序层次详细信息，参考本手册ISaGRAF概述章节，或ISaGRAF用户指南。

然后 →

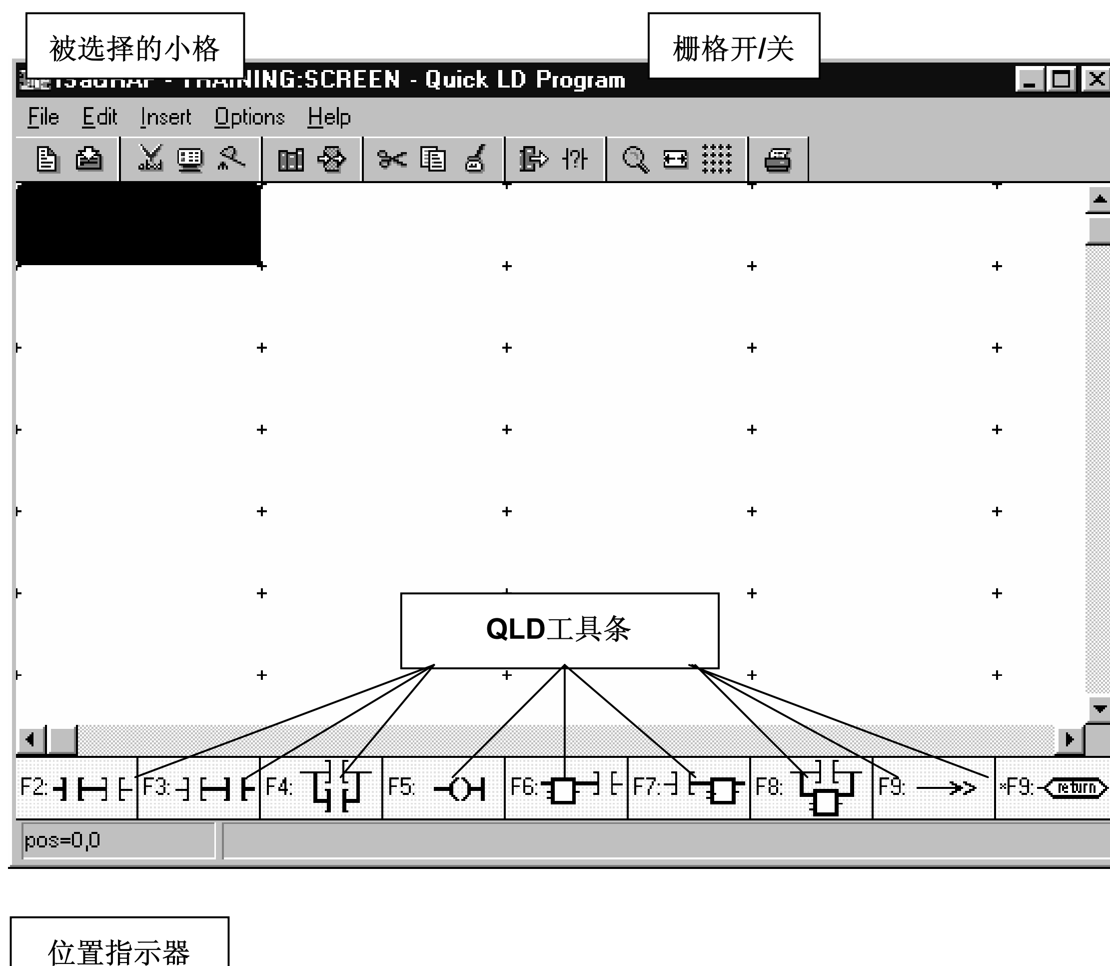
双击 **RunMotor** 编程图标，进入QLD（快速梯形图编程）环境。



快速梯形图编辑器环境：

ISaGRAF快速梯形图编辑器（Quick LD）可以使用键盘和鼠标容易地建立梯形图程序。它按照逻辑栅格自动地连接和布置阶梯。

本培训假定大多数编程由鼠标完成。关于功能键和键盘的用法请参考ISaGRAF用户指南。

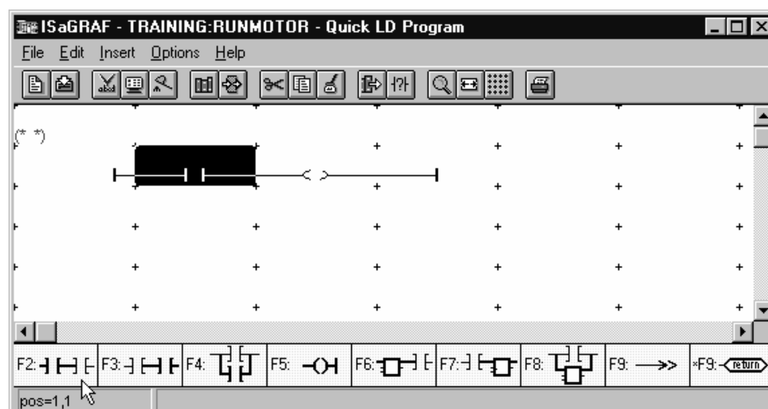


然后 →

点击 F2 图标，插入第一个阶梯。

注: ISaGRAF 自动将左栏杆放置到（1，1）点，并自动插入接点、线圈和右栏杆。

(点击 图标可加上栅格).



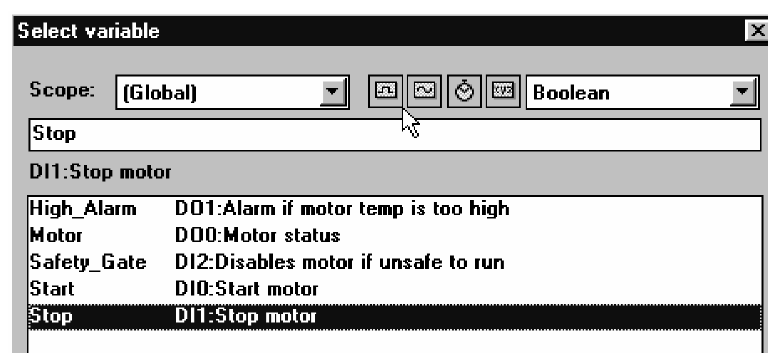
然后 →

双击接点小格**(1,1)**，以给接点分配一个变量。

然后 →

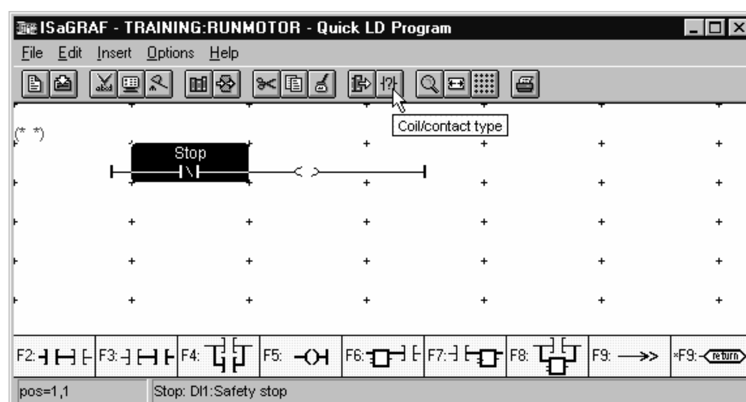
选择 **Stop** 并点击 **OK.**

(如果没有显示出布尔变量，点击按钮 即可列出。)



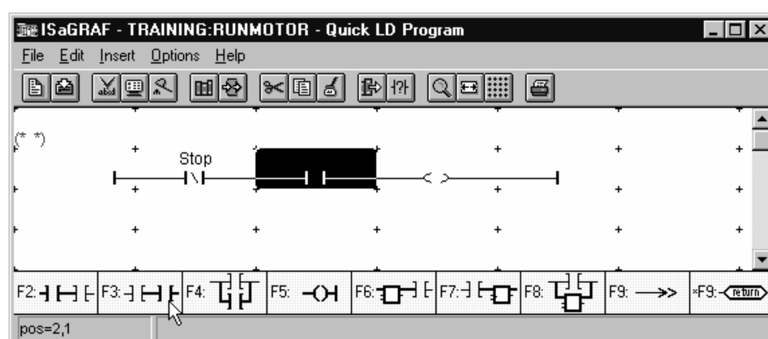
然后 →

点击 ，将 **Stop** 设为常闭变量接点 (-|\|-) 。



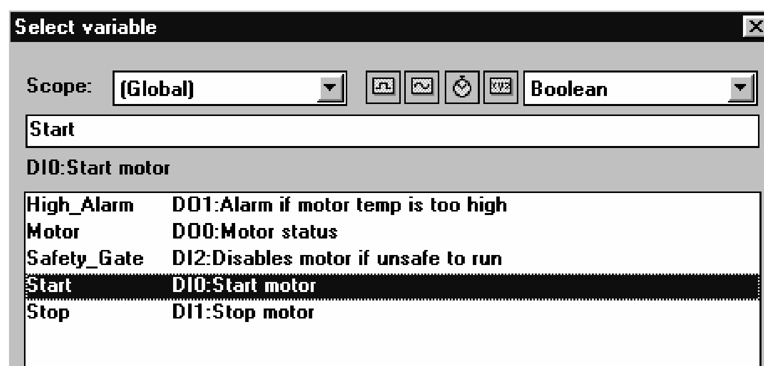
然后 →

点击 F3，插入新接点。



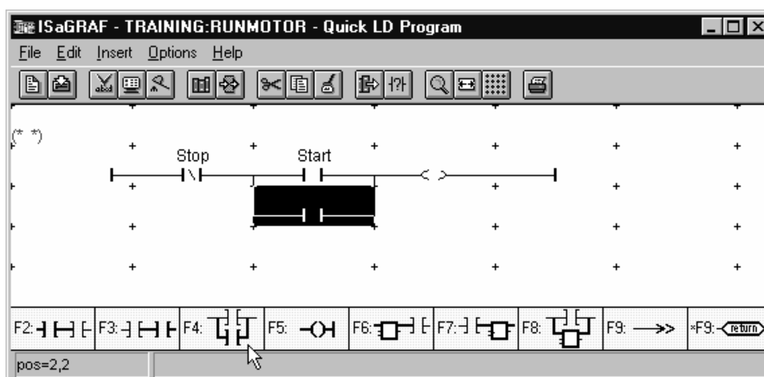
然后 →

将**Start**变量分配给它**(2,1)**。



然后 →

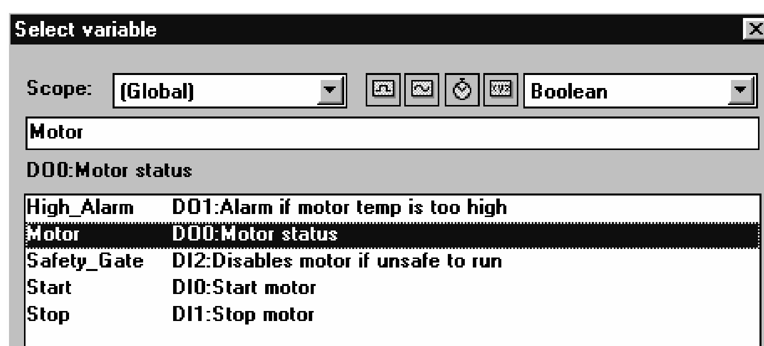
插入平行接点，选择小格(2,1)，然后点击 F4 图标，一个平行接点将自动放置到小格(2,2)。



然后 →

双击小格**(2,2)**，将变量**Motor**分配到该接点。

从**Select Variable**窗口，选择**Motor**并点击**OK**.



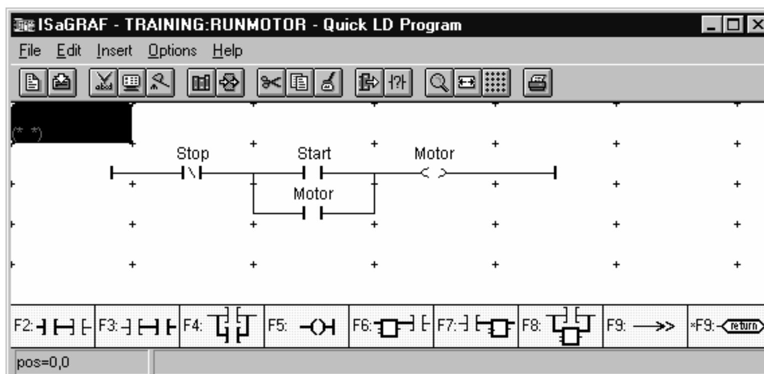
然后 →

双击小格(3,1)，将**Motor**变量分配到线圈。

然后 →

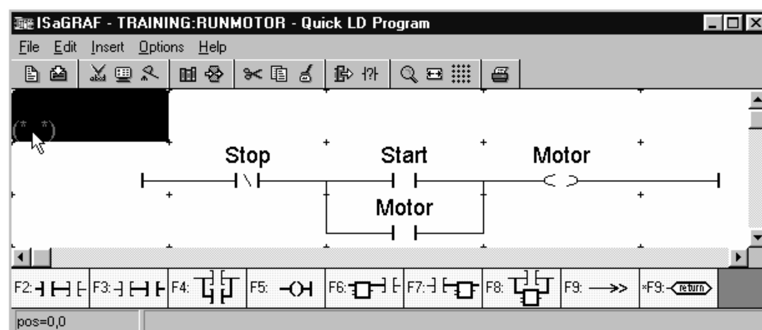
从**Select Variable**窗口，选择**Motor**并点击**OK**.

程序的基本轮廓就完成了。快速梯形图编辑器已完成了所有的连接。



您的程序看起来应是这样的

注意：小格(0,0) 显示的**(* *)** 是一个注释栏，可以双击它输入简单的程序说明。

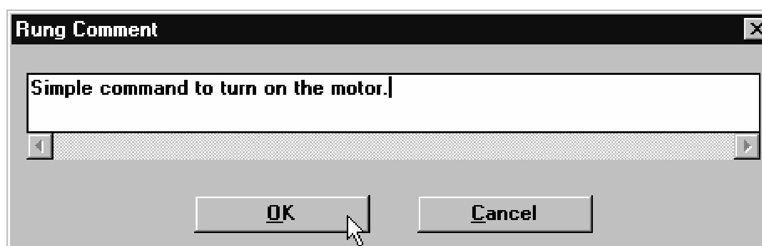


然后 →

双击小格(0,0)处的**(* *)**，写入：
Simple command to turn on the motor.

然后 →

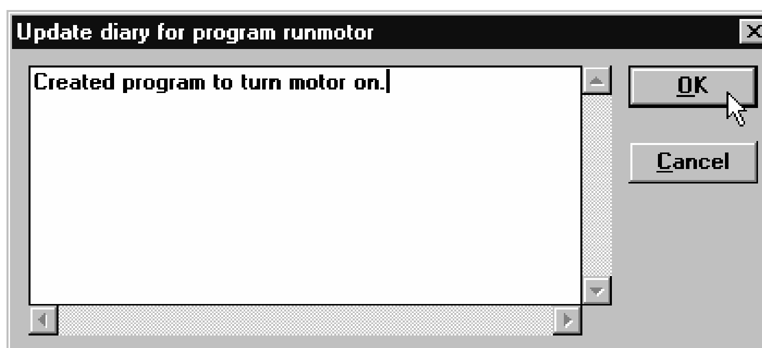
点击 **OK.**



然后 →

然后 →

点击 图标，保存该程序。



这时，日记工具窗口出现，它允许您在每次修改程序时做出记录，请写入适当信息，点击 **OK.**

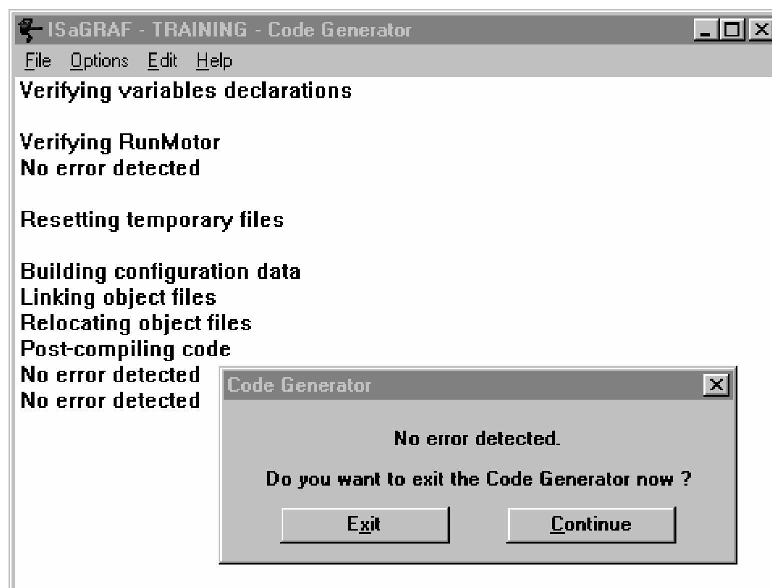
然后 →

点击 **OK.**

然后 →

点击 (确认)图标，检查程序错误，如有错误，则用红字表示出来。

双击该字，可进入出错处。



然后 →

当程序被确认并无错时，退出代码生成器。

关闭确认窗口，返回 **RunMotor** 程序。

如果确认时有错误，双击该红色错误信息，ISaGRAF将带您到出错处。修改差错，重新保存和确认之。

程序中加入功能块：

现在我们要利用加入功能块实现安全性能：监视马达温度，当其过热（高于200℃）时，发出报警声，并关闭马达。

然后 →

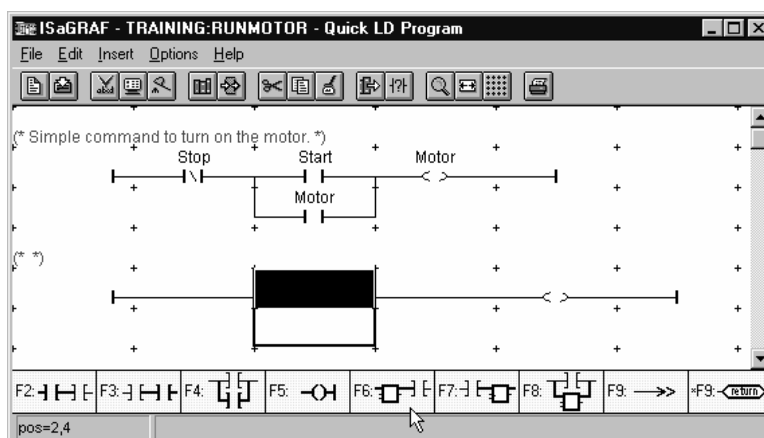
在 **RunMotor** 程序中选择小格(0,4)。

然后 →

点击 F6: 图标。

然后 →

双击功能块中心，将弹出功能块列表。



然后 →

从功能块列表中选择 **>=** 功能块并点击 **OK** 。

注意: En (使能) 自动加在功能块左上角，在梯形图程序中，功能块必须接受一个布尔输入，以激活（使能）它。En为真时，功能块激活。

由于 **Motor_Temp** 原始值为 0-32767，而我们报警是以200℃，所以应创建一个内部变量(scale_temp)以将Motor_Temp表示为工程单位。

然后 →

点击 图标（变量字典）。

然后 →

点击 图标，列出模拟变量（**Integers/Reals**）

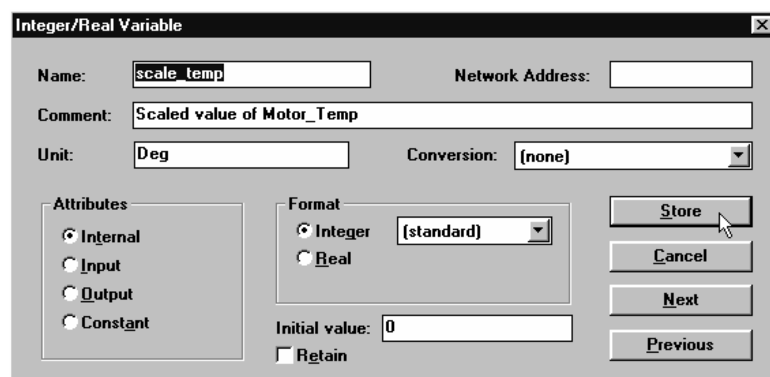
然后 →

选择 **Edit, New.**

然后 →

按下列参数填写弹出窗口：

Name: scale_temp
Comment: Scaled value of Motor_Temp
Unit: Deg
Attribute: Internal
Format: Integer



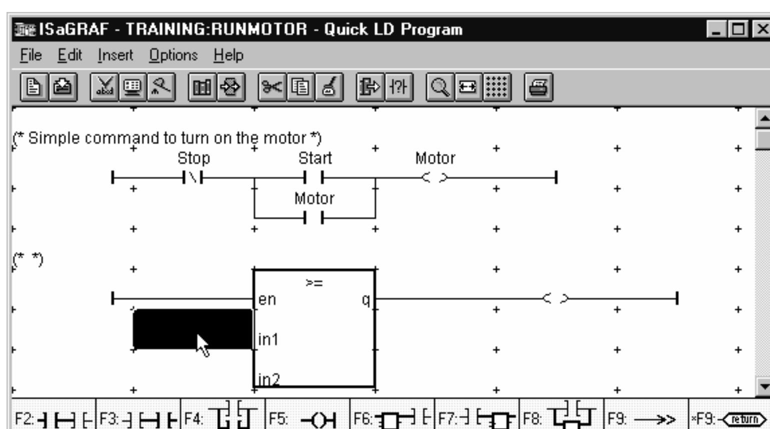
然后 →

点击 **Store** ， 然后 **Cancel** 。

选择 **File, Exit** 及 **Save** 。

然后 →

双击小格(1,5)。会显示出布尔(离散)值来。

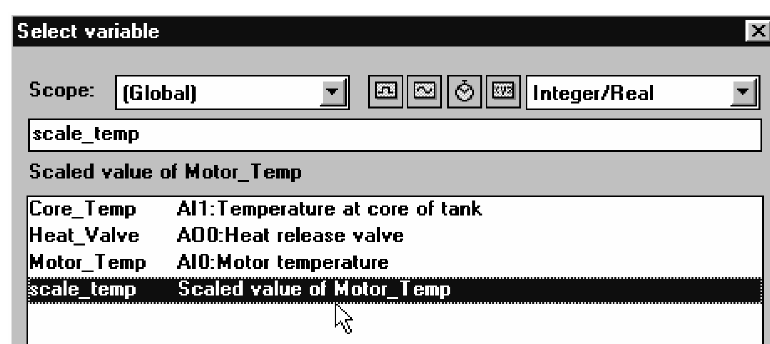


然后 →

点击 图标，以显示整形/实型变量。

然后 →

选择**scale_temp**变量，点击**OK**。将它分配到功能块的第一个输入上。



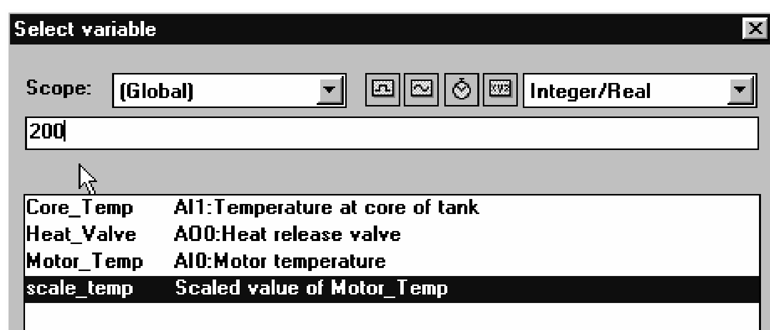
然后 →

双极小格 **(1,6).**

应显示出整形/实型变量。

然后 →

写入 **200.** 点击**OK.**



现在，功能块作用是：

如果 **scale_temp** 等于或大于 **200**，则输出为真。

然后 →

双击阶梯的线圈格(4,4)。

然后 →

点击 图标，显示出布尔变量。

然后 →

选择**High_Alarm**并点击**OK**，将其分配给块的输出。

然后 →

光标移到**Stop**接点(1,1)上。

然后 →

点击图标 F2 在**Stop**左面插入一个接点。

然后 →

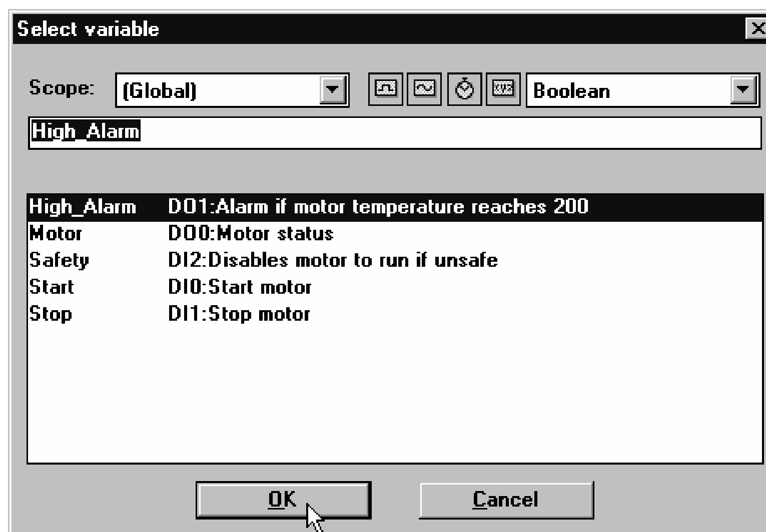
双击这个新的接点。

然后 →

选择 **High_Alarm**， 点击 **OK**.

然后 →

点击 图标，选择常闭接点给 **High_Alarm**。

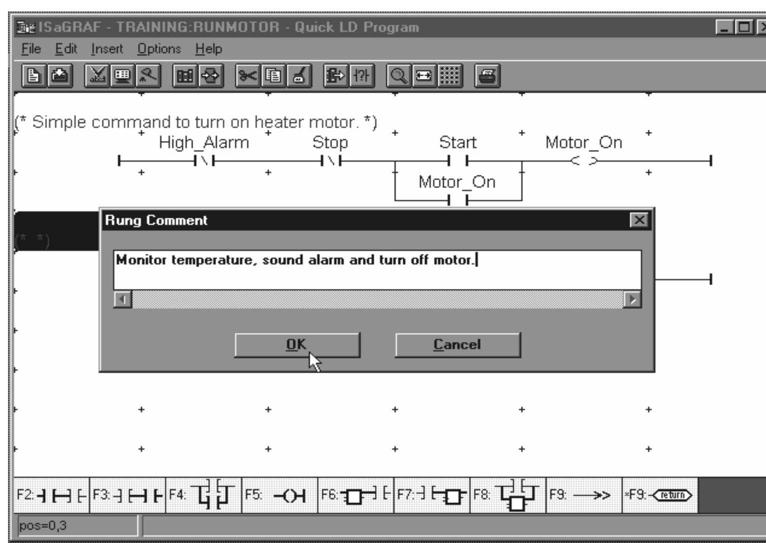


然后 →

双击格(0,3)的 **(* *)**， 写入：***Monitor temperature, sound alarm, turn off motor.***

然后 →

点击 **OK** 。



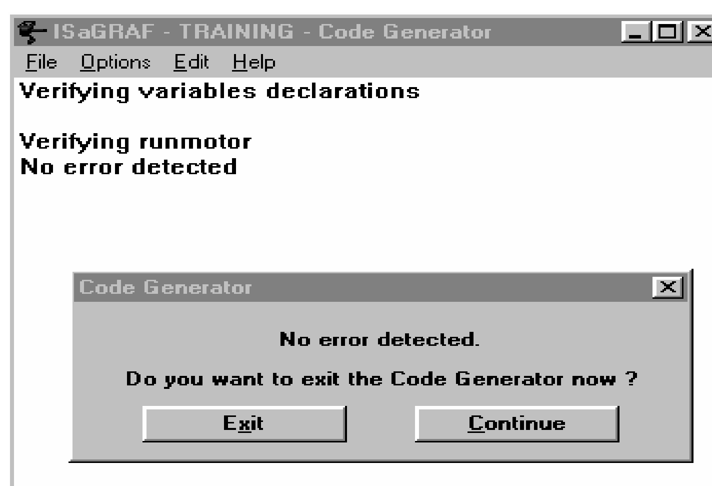
然后 →

保存并确认该程序（**Save / Verify** ）

然后 →

在程序确认为无错后，退出代码生成器。（**Exit**）

关闭确认窗口，并关闭**RunMotor**程序窗口。



现在，我们将准备学习下一章 *快速梯形图/功能块图编辑器”*

本章我们将要：

- 学习梯形图/功能块图（LD/FBD）编辑器
- 使用LD/FBD编辑器建立两个程序
- 保存和确认程序



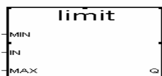
关于编辑器的详细信息，请参考 ***ISaGRAF*** *用户手册的* ***LD/FBD*** *编辑器使用章节或在线帮助*

开始本章前，您应当：

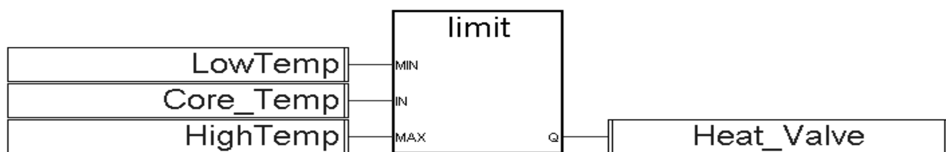
完成了本手册的概述章，以及*快速梯形图编辑器使用*章。

功能块图编程基础：

一个功能块图(FBD)描述了输入变量和输出变量间的一个函数，执行不同操作的功能块表示为一个图形集，功能块图由名称、输入参数、输出参数组成为一个长方形图符，如：



输入和输出变量由连接线连到块上，并应与要求的数据类型一致(Boolean, integer, real, etc.)。输入连到左边，可以是常数、内部变量、输入变量、输出变量；输出连到块的右边，应为内部或输出变量。



与梯形图一样，功能块图总是从左向右、从上到下解释。可以将功能块图与梯形图一起使用

关于功能块图及梯形图编程的详细信息，请参考 ***ISaGRAF*** *语言参考手册*

实际练习：

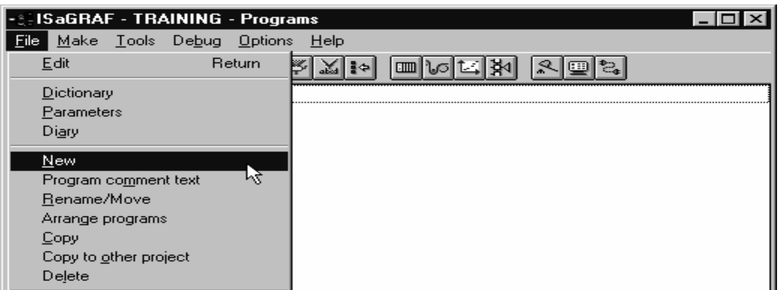
本练习中，我们将建立两个简单的功能块图程序。

第一个程序（Scale），将原始模拟输入值 **Motor_Temp** (1 - 32767) 按比例转换为工程单位值(1 - 300℃)，结果赋给内部变量 **scale_temp**（在QLD章中创建）。

第二个程序（Safety），是一个安全控制，检查目标状态以确保控制安全。

SCALE程序开始：

从Training程序窗口，选择 File, New.



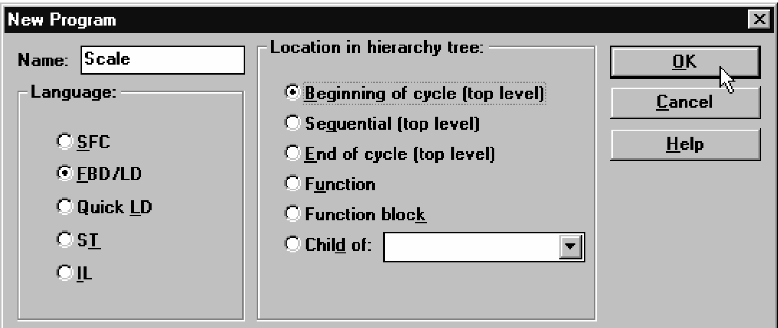
然后 →

按下列参数创建Scale程序：

Name : **Scale**

Language: **FBD/LD**

Location: **Beginning of Cycle**



点击 **OK**.

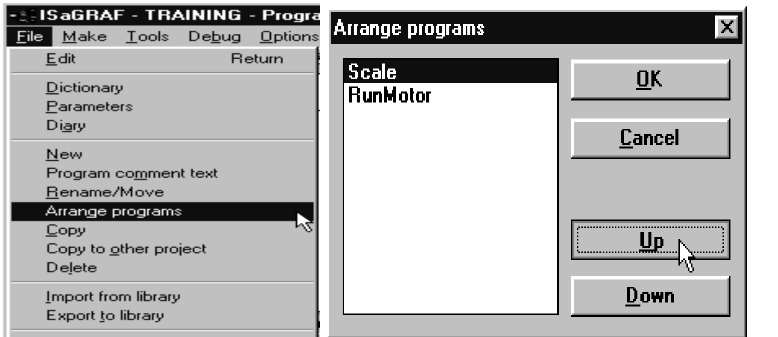
说明：在ISaGRAF目标循环中，为使RunMotor程序使用scale_temp变量的当前值，Scale程序应放到RunMotor程序的前面。

然后 →

选择 **File**, **Arrange Programs**.

选择 **Scale** 并点击 **Up**.

点击 **OK.**



然后 →

双击 Scale 程序。

当移动鼠标时，可观察屏幕左下角的位置指示。

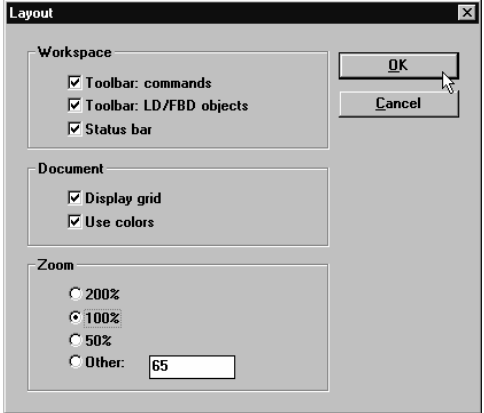
然后 →

选择 **Options** 并确保选中 *FBD toolbar* 和 *Auto input* .

然后 →

选择 **Options, Layout** 并确保选中所有的 **Workspace** 和 **Document** 选项。





将1 ~ 32767的原始值按比例转换为整形值1 ~ 300：

$$\frac{\text{原始值} \times 300}{32767} = \text{比例转换值}$$

Scale程序将原始的Moter_Temp值转换为工程单位 (℃):

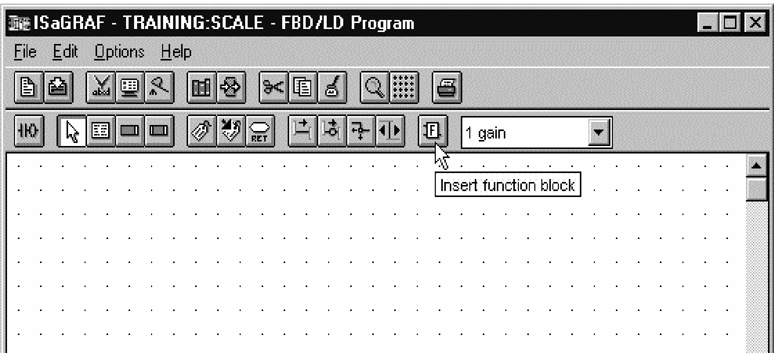
$$\frac{\textbf{Motor_Temp} \times 300}{32767} = \textbf{scale_temp}$$

然后 →

点击工具栏上 按钮。

然后 →

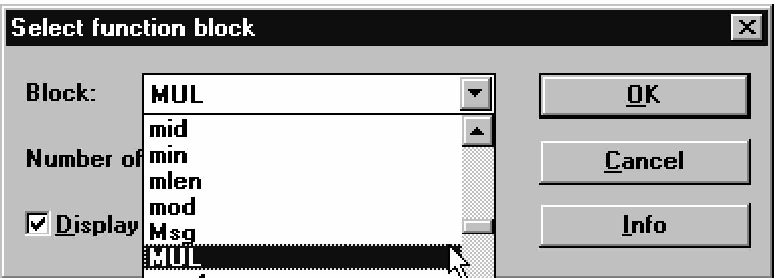
双击小格 (12,2).



然后 →

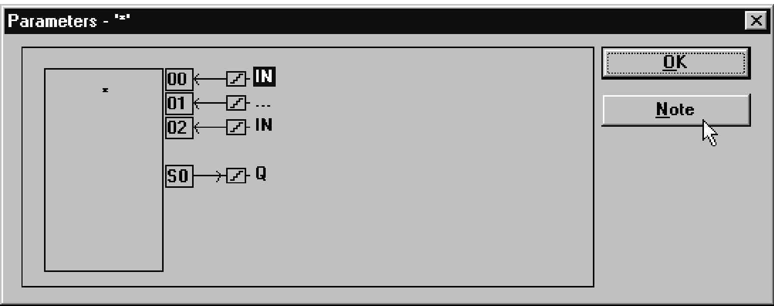
在功能块选择窗口，选择 MUL 功能块。

注意：输入变量的数目可选，缺省为2。



然后 →

点击INFO看该块的结构，及输入/输出参数数据类型。



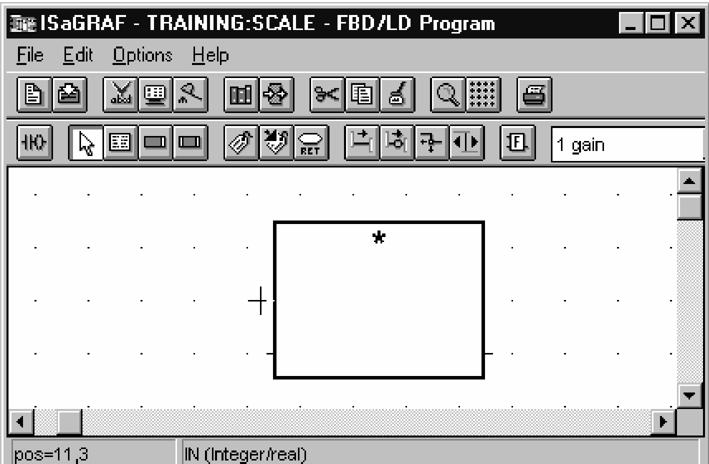
然后 →

点击Note看该块的详细信息。

然后 →

将鼠标放到功能块的第一个输入脚左边。

此时IN (Integer/real) 出现在屏幕左下角位置标志的右边，表示该输入所要求的类型是实型/整型。

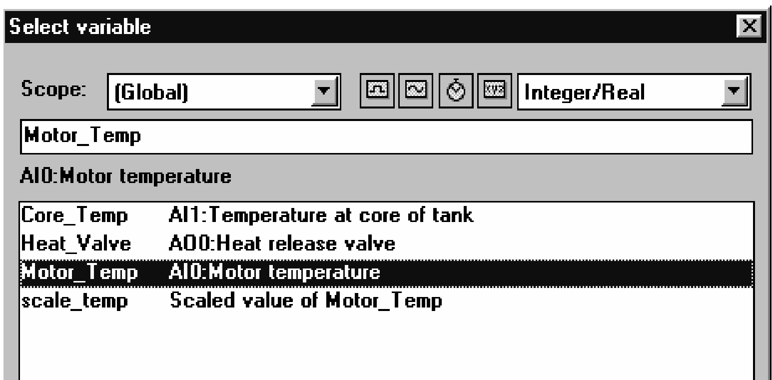


然后 →

点击 图标，并点击小格(1,3).

然后 →

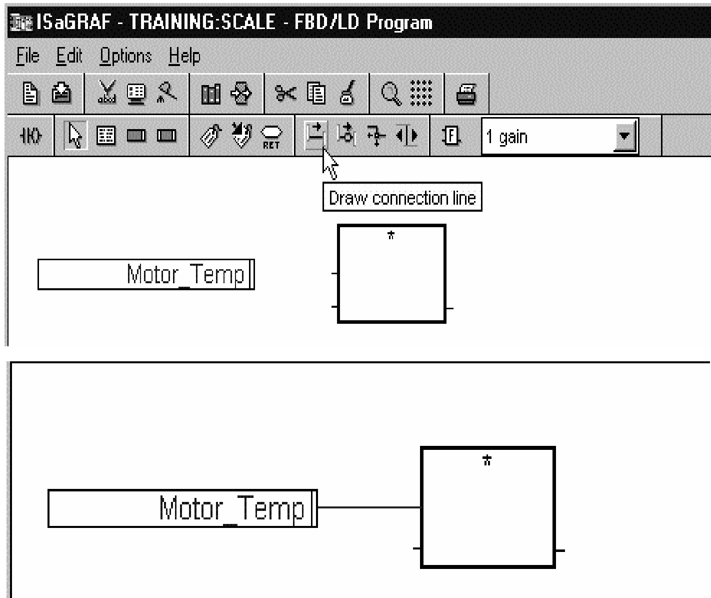
在变量选择窗点击 图标，选择变量 Motor_Temp，点击OK。



然后 →

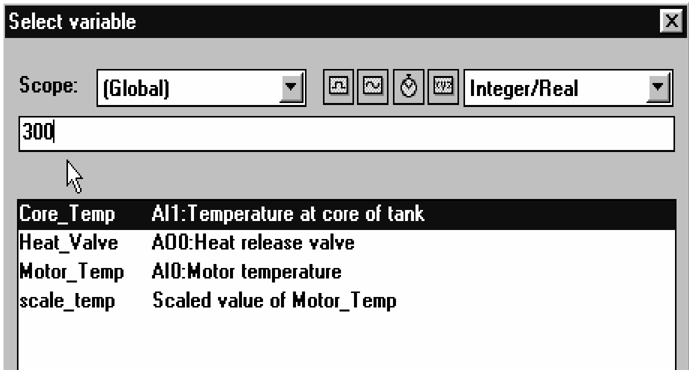
连接变量 MOTOR_TEMP 到功能块第一个输入：

点击 图标（画连接线），在变量右侧按下鼠标左键，将鼠标拖到MUL功能块的第一个输入脚。



然后 →

点击 按扭及小格(1,4).

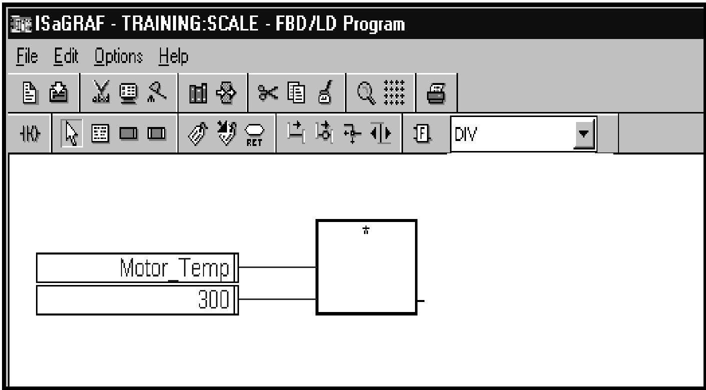


然后 →

在变量选择窗口直接写入缺省变量300. 点击 OK.

然后 →

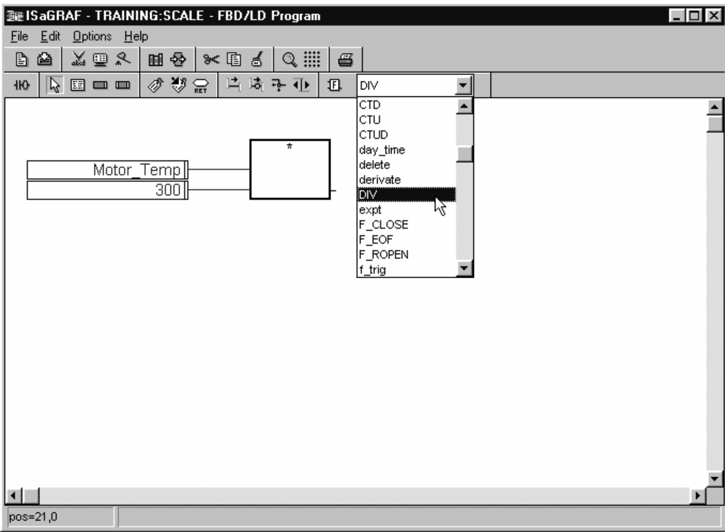
选择 按扭，将变量连接到MUL功能块的另一输入脚。



该功能块的输出为变量 Motor_temp 与 300 的乘积。该输出应被32767除，才可返回比例后的值。

然后 →

点击 按钮加入DIV（除法）功能块。



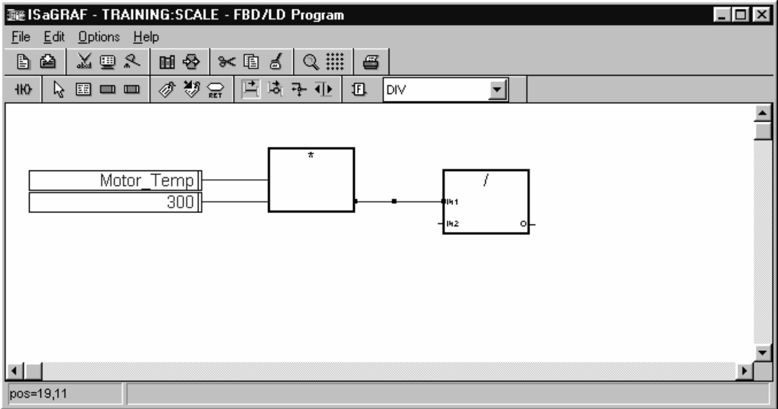
然后 →

不从弹出框，而是使用下拉菜单选择功能块。

然后 →

选择DIV功能块，并点击小格 (20,3) *一次*。

DIV功能块用第二个输入去除第一个输入。

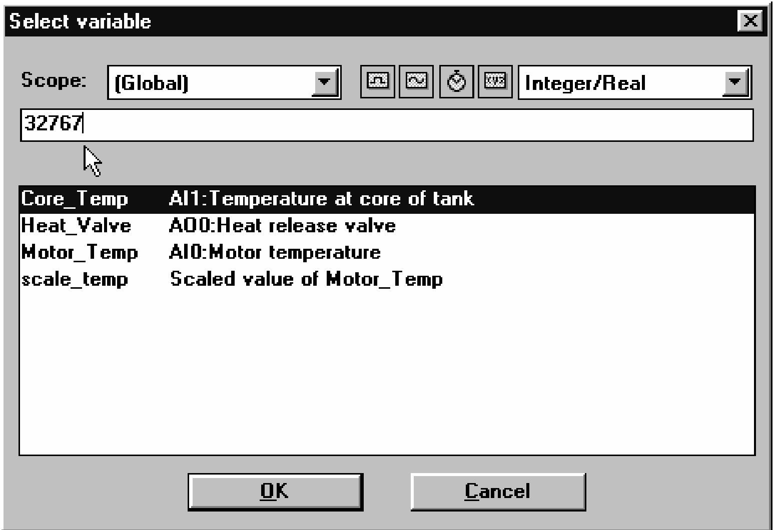


然后 →

使用 按钮连接MUL功能块的输出到DIV块的第一个输入，小格(16,4)连接到 (20,4).

然后 →

点击按钮 ，点击小格 (10,5).



然后 →

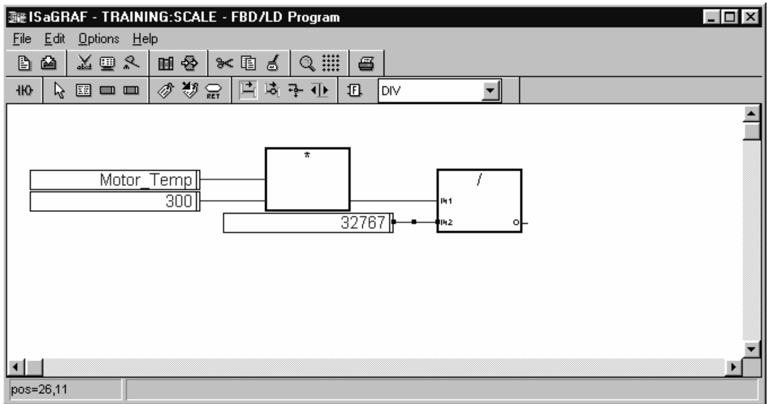
在变量选择窗口，输入 **32767.**

然后 →

点击 **OK**.

然后 →

使用 按钮将32767连接到DIV块的第二个输入， (18,5) 到 (20,5).

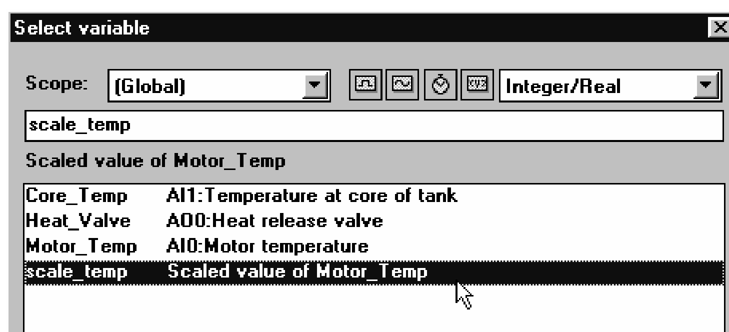


然后 →

点击 按钮和小格 (28,5).

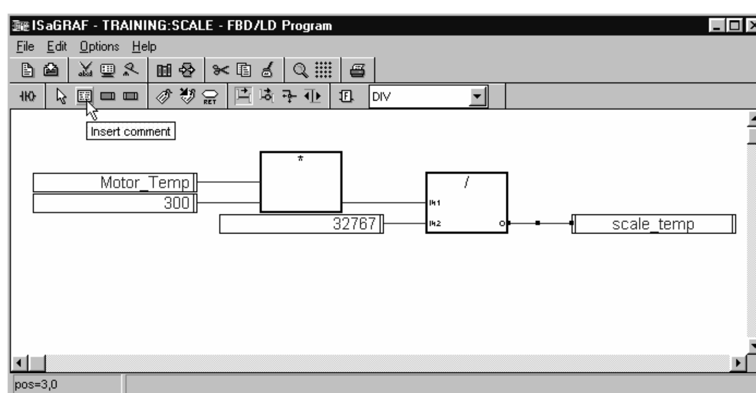
然后 →

在变量选择窗口，选择 scale_temp. 点击 **OK**.



然后 →

使用按钮 将 **scale_temp** 连接到DIV功能块的输出，即(24,5) 连到(28,5).

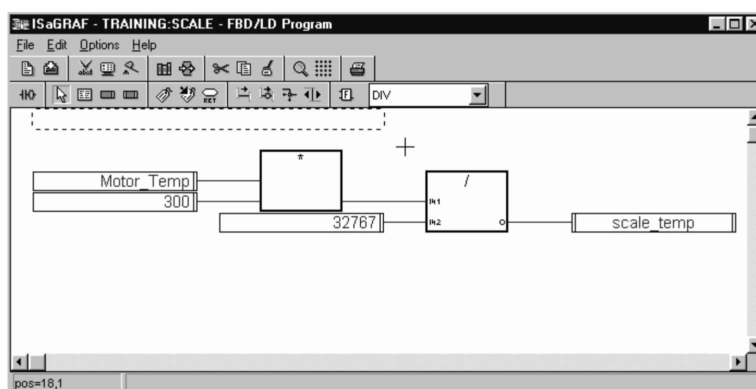


然后 →

点击 按钮插入程序注释，光标移至(1,0)。

然后 →

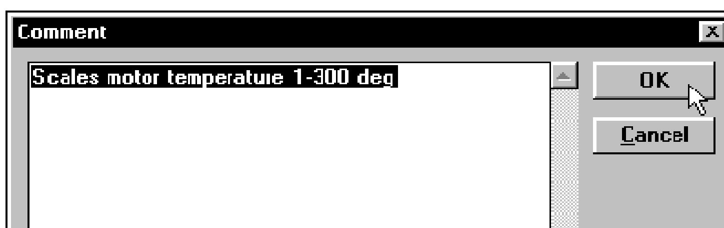
保持鼠标左键按下，并将鼠标拖至小格 (20,1)，然后放开。



然后 →

写入注释: 揝**cales motor temperature to value between 1 and 300”**

点击 **OK**.

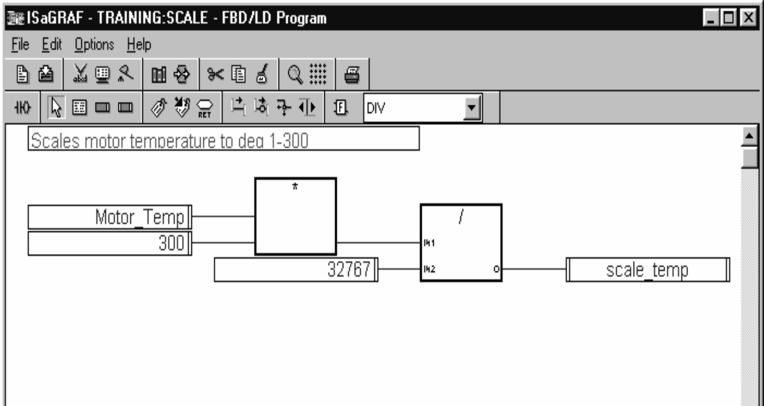


然后 →

保存并确认该程序。

然后 →

当确认无错误后，退出代码生成器。关闭确认窗口及Scale程序窗口。



如果您没有完成本手册使用快速梯形图编辑器章，请参考本章第8页如何确认程序。

SAFETY程序：

本程序目的是：检查在写入任何输出前，接点Safety_Gate是关闭的。如果Safety_Gate是开的，马达就关上。把此程序放到ISaGRAF目标循环结束部分，这样即使RunMotor程序执行ON命令时，也不对马达操作。

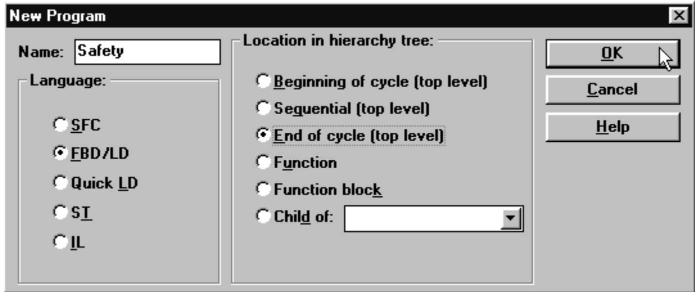
然后 →

选择: **File, New Program**

程序名: **Safety**

语言: **FBD/LD**

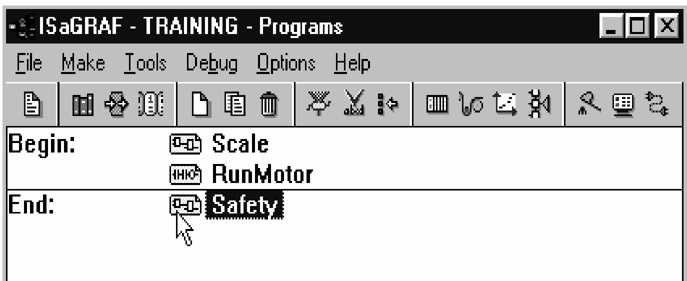
位置: **End of cycle**



然后 →

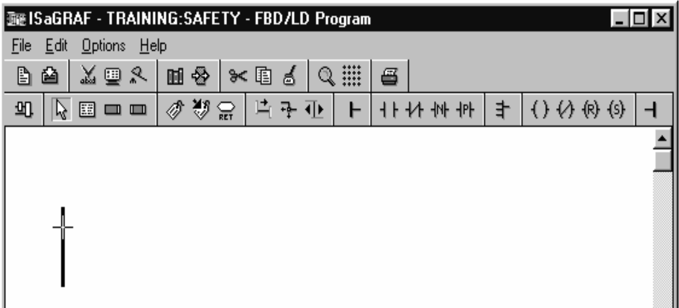
打开Safety程序，并确保Options菜单下的Auto-input选项选中。

点击 [图标] 图标，显示出梯形图工具。



然后 →

选择 [图标] 图标（左栏杆）并将其放置到小格(2,3)：点击小格。

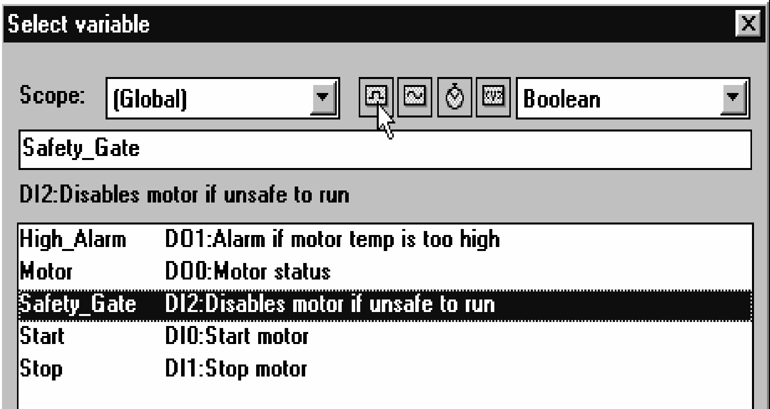


然后 →

选择 图标（常闭接点），点击位置 (4,4)。

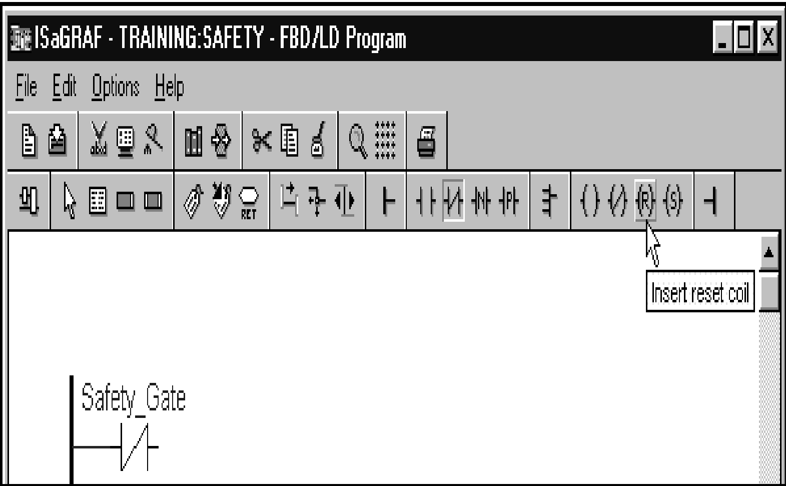
然后 →

从字典中选择Safety_Gate变量（布尔类型）。



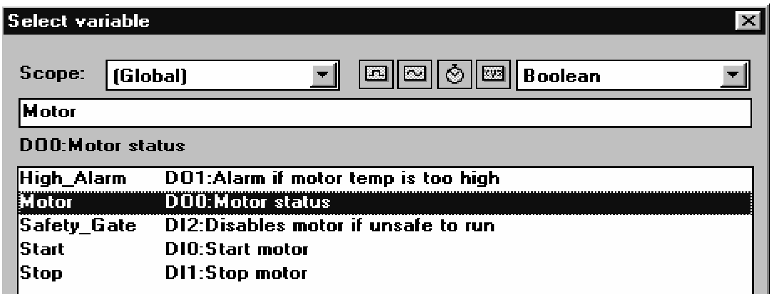
然后 →

选择 图标（复位线圈），点击位置 (9,4)。



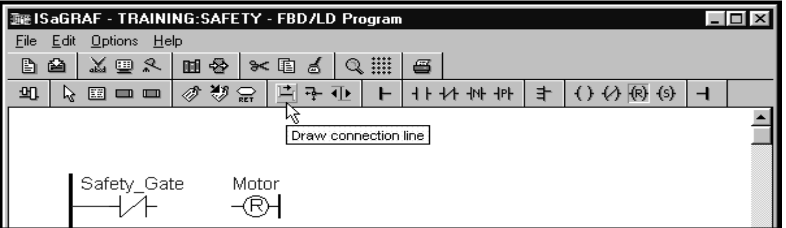
然后 →

从字典中选择Motor变量（布尔类型）。



然后 →

选择 图标（连接）将接点和线圈连起来。

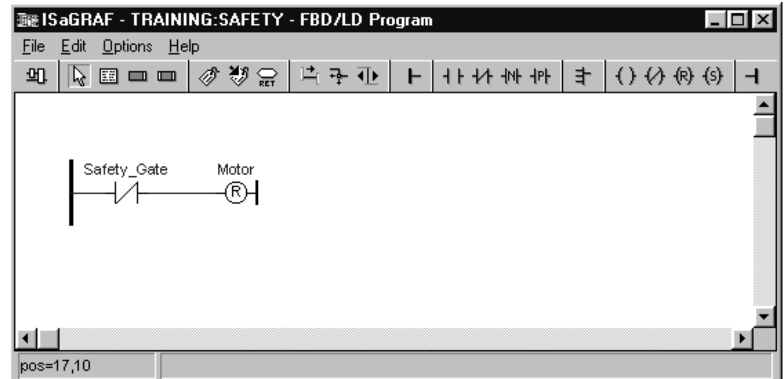


然后 →

现在您的程序逻辑按如下工作：

如果**Safety_Gate**是关闭的，则开关保持关闭。常闭接点上不通过电源，没有动作。

如果**Safety_Gate**是打开的，电源通过马达线圈并复位它。由于Safety程序在每个执行循环的最后执行，任何在完成循环更新输出前的马达动作将被阻止。



然后 →

保存并确认该程序。

然后 →

确认无错误后，退出代码生成器。

关闭确认窗口，关闭Safety程序窗口。

本章我们将要：

- 设置编译器选项
- 编译应用
- 学习如何给应用加上密码，以提高安全性



关于编译的详细信息，请参考 ISaGRAF用户手册或在线帮助

开始本章前，您应当：

已完成*使用**Quick Ladder**编辑器*和*使用**LD/FBD**编辑器*章节，已创建了 ISaGRAF **TRAINING** 项目的程序。

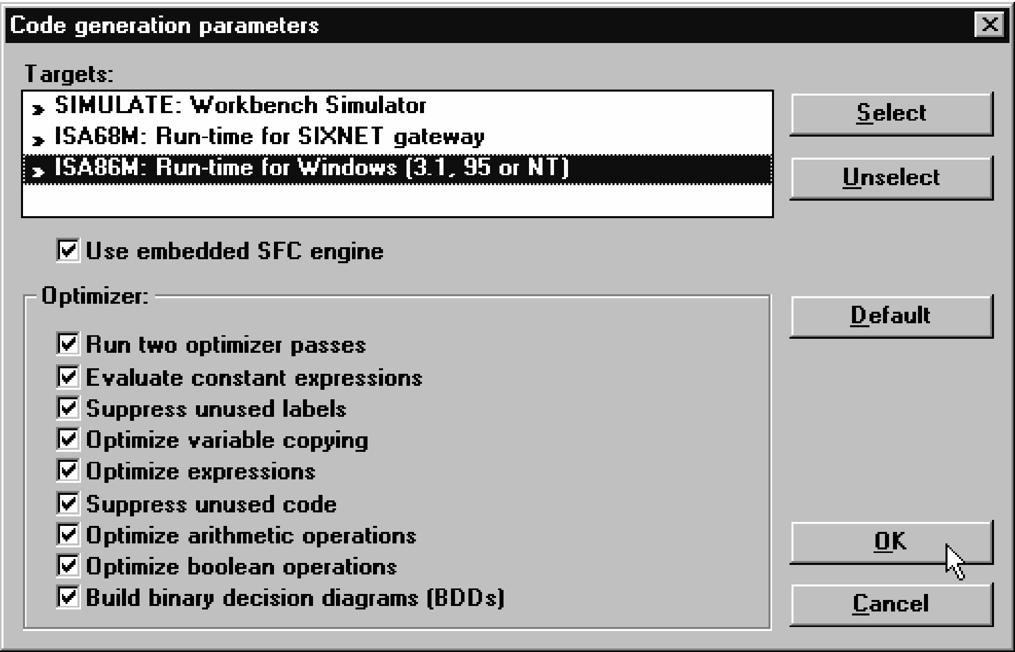
编译ISAGRAF应用：

Make菜单的命令用于运行代码生成器，及输入选项和附加数据。运行命令前必须正确设置目标代码生成选项。

生成目标代码前，所有未经确认的程序将先进行语法差错检查。ISaGRAF包含一个增量编译器，它对上次编译过的、未修改的程序不再进行编译。

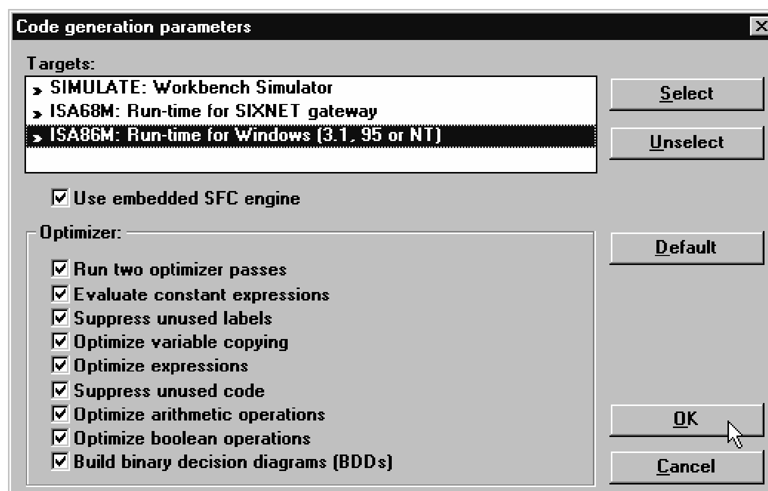
开始：

关于编译器高级选项，请参考 ISaGRAF 用户手册



从ISaGRAF Training程序窗口：

选择 **Make, Compiler Options**.



注意编译器选项窗口的三个目标选项：

- **SIMULATE: Workbench Simulator –** 在ISaGRAF的仿真器上运行(不要求硬件)。这是您在办公室调试程序的好办法。
- **ISA68M: Run-time for SIXNET gateway** – 在VersaTRAK或SIXTRAK可编程控制器中运行程序。
- **ISA86M: Run-time for Windows (3.1, 95 or NT) –** 使用SIXNET Control Room 和 ISaRUN run-time 软件，在基于**Windows** 的 **PC** 上运行程序。

每个选项将按照其目标处理器编译程序。ISaGRAF代码生成器可在同一次编译时产生所有的三种版本代码。

然后 →

使用**Select**按钮选择所有三个选项：**SIMULATE**, **ISA68M** 和 **ISA86M** 。

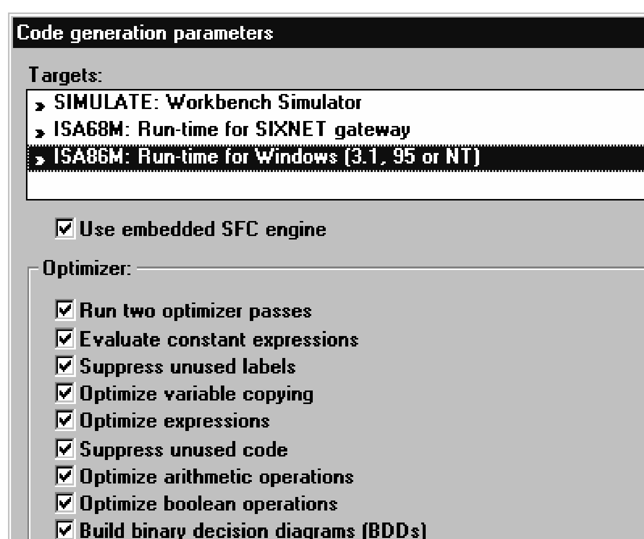
然后 →

选择所有的优化选项。

选择 **OK**.

会通知您下次编译时所有程序将被确认。点击 **OK**。

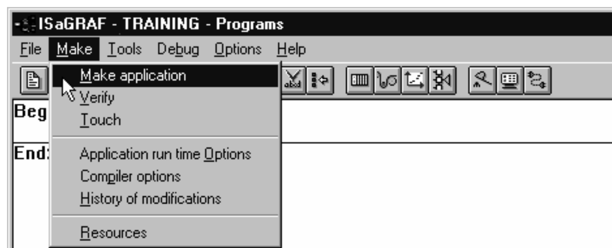
现在将可以在所有三种目标中运行您的应用。



然后 →

选择 **Make, Make Application.**

ISaGRAF代码生成器将确认程序，无误后编译之。

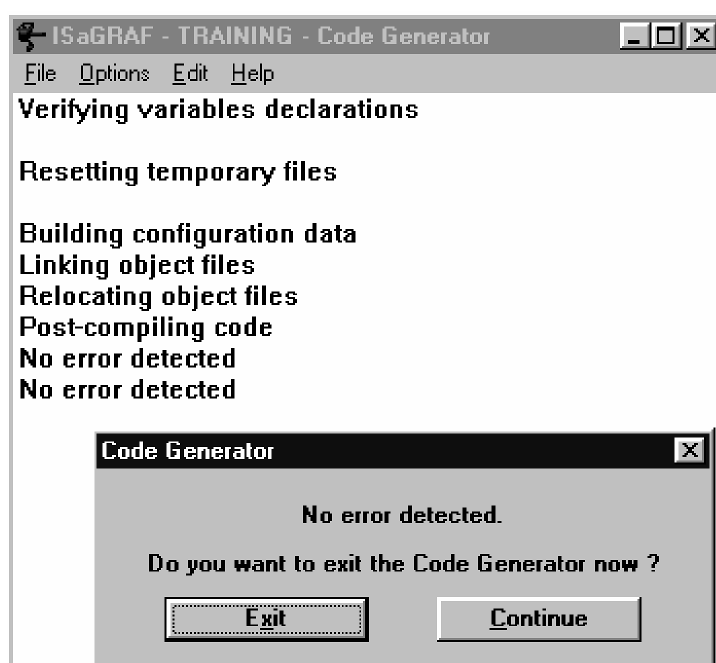


然后 →

选择 **Exit**.

然后 →

关闭training程序窗口。



使用密码保护TRAINING项目

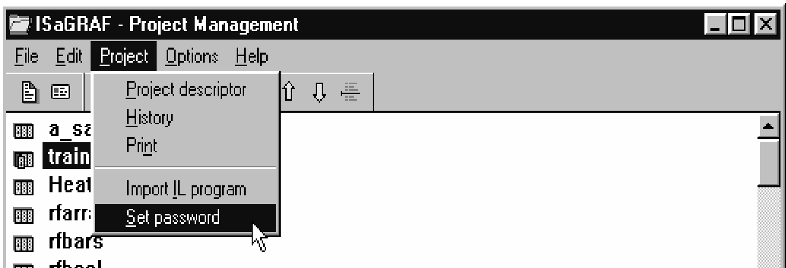
ISaGRAF为保护应用提供了16个保护级别。 密码级别从0至15，0是最高级。

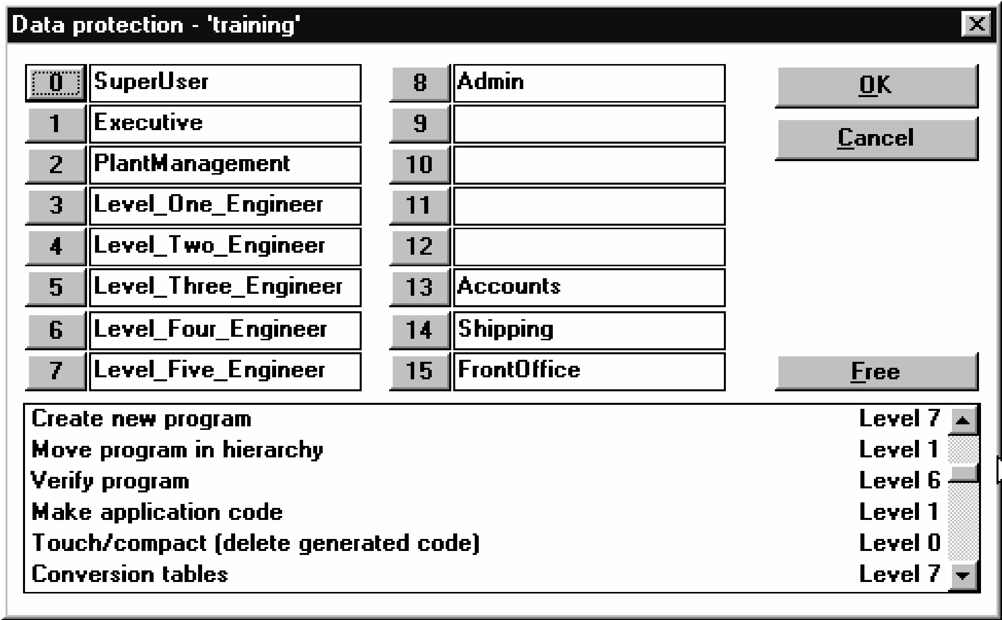
从**ISaGRAF**项目管理窗口：

选择**Training**程序 (选中但不打开).

然后 →

选择 **Project, Set Password.**





在相应密码级别号码的右边文本框，按您的要求写入密码。

窗口下半部分是项目的元素列表 (如：创建新应用，编译应用，移动程序等)。选择各元素，然后点击要求的级别，可分配访问级别。

选择元素，然后点击 **FREE** 按钮，可去掉保护密码。退出ISaGRAF，然后重新打开项目管理器，密码才可生效时。

注：TRAINING应用不需要密码保护。

然后 →

选择**Cancel**，退出密码窗口。

本章我们将要：

- ISaGRAF仿真器概述
- 运行I/O仿真器调试RunMotor程序
- 使用仿真器调试Safety程序



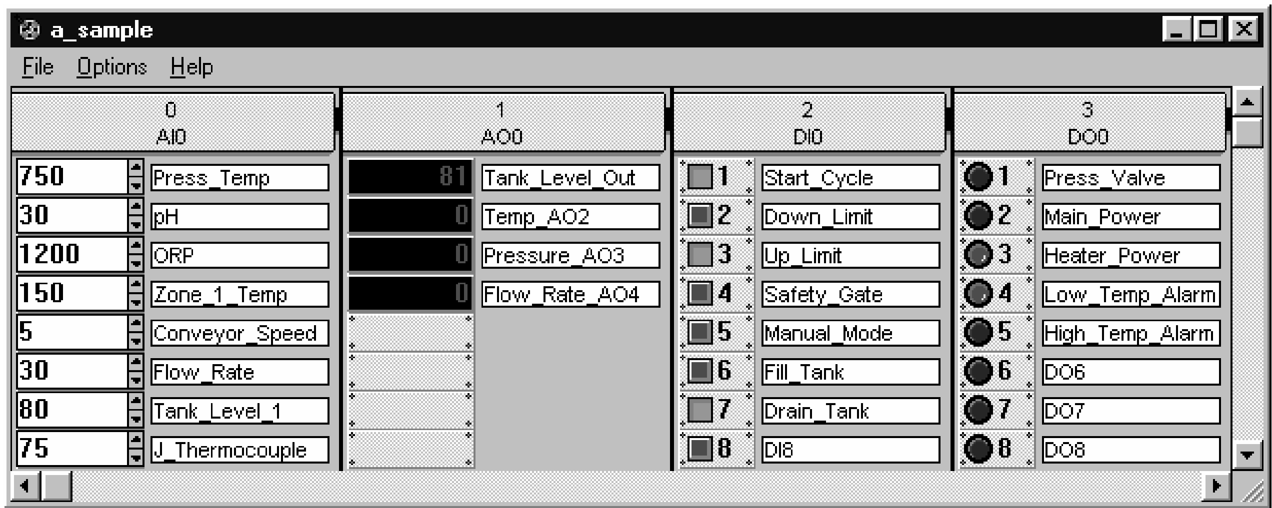
关于ISaGRAF仿真器的详细信息，参考ISaGRAF在线帮助或用户手册

本章开始前，您应当：

已完成了本手册的使用梯形图/功能块图编辑器章节。

I/O仿真器：

ISaGRAF I/O仿真器是一个完整的ISaGRAF目标系统，支持ISaGRAF标准特性和CJ International和SIXNET发布的的所有标准库函数和功能块。仿真器支持和ISaGRAF调试器的完全通信，所以仿真时具有所有的调试功能。*如果在上次编译后做过修改，则运行仿真器前应重新编译。*



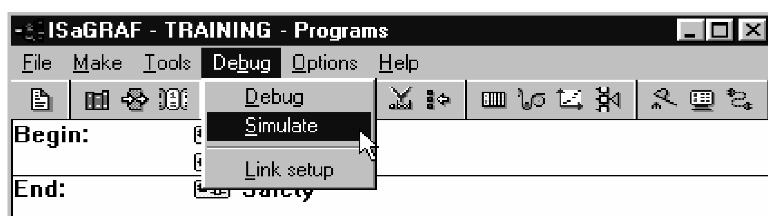
开始：

打开Training项目。

然后 →

从ISaGRAF Training 程序窗口:

选择 **Debug**, **Simulate**.

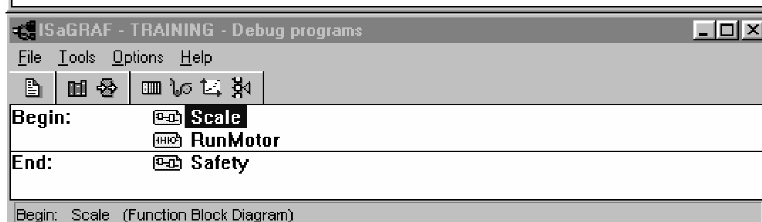


当仿真器启动时，调试器，Training 程序，及仿真器窗口以调试方式打开。

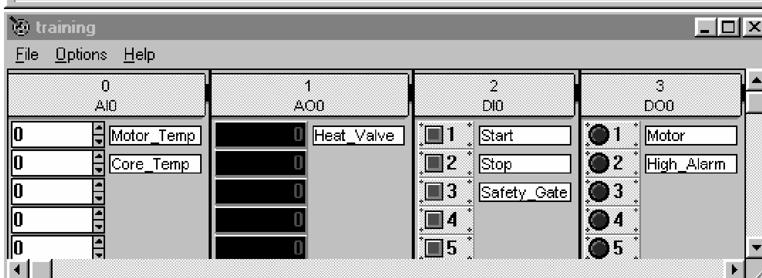
调试器窗口 →



调试方式下的程序窗口 →



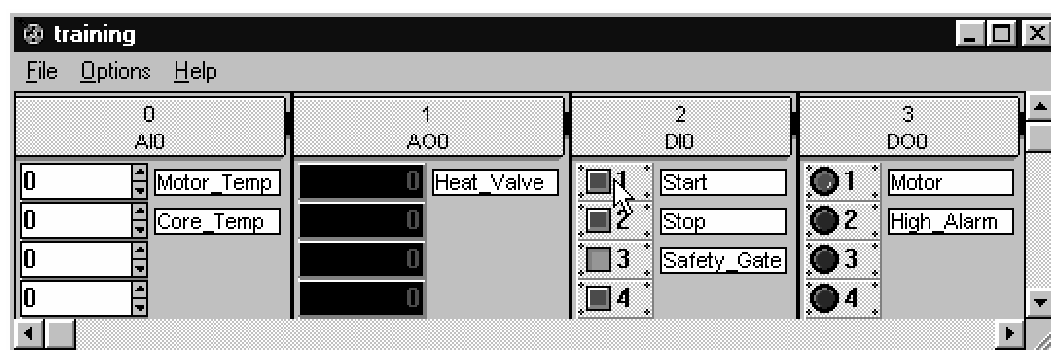
ISaGRAF I/O 仿真器 →



然后 →

如果您的窗口与右边所示不同，选择 **Options**, 确保之选中：**Color display, Variable names, Always on top.**

> *关于如何使用调试器，请参考本手册使用调试器章，或查看 ISaGRAF用户指南*



显示出*Training*项目的各个I/O点

Analog Inputs: (最左边所示) 点击输入点数值框，输入原始数值。也可使用[上下箭头]按钮增加或减少数值。

Analog Outputs: (本例没有) 输出值以红色数字表示。

Discrete Inputs: (右边第二栏) 鼠标左键就象一个触发开关 – 按下为真(亮)，弹起为假(灭)。鼠标右键就象一个按钮 – 按钮按下时输入为真。

Discrete Outputs: (最右边栏) 由红色LED表示的输出。输出为真时LED亮。

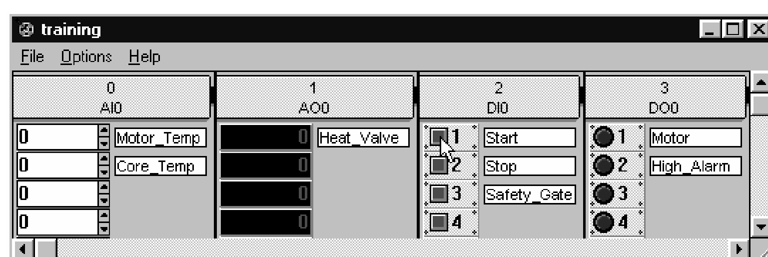
使用仿真器测试应用：

开始：

左击 **Safety_Gate** LED，使能马达操作。

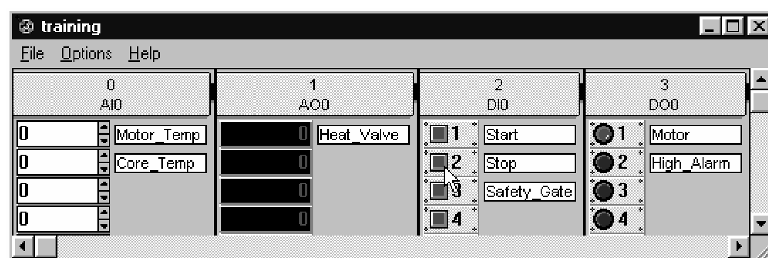
然后 →

右击 **Start** 按钮启动马达。注意现在马达**LED**为亮(真)。



然后 →

右击 **Stop** 按钮。注意现在马达**LED**为灭(假)。



然后 →

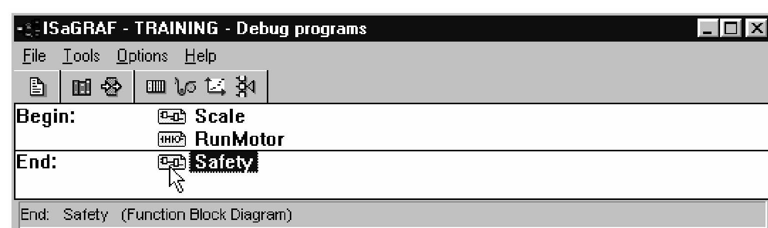
重新右击 **Start** 按钮。

然后 →

双击 **RunMotor** 程序。

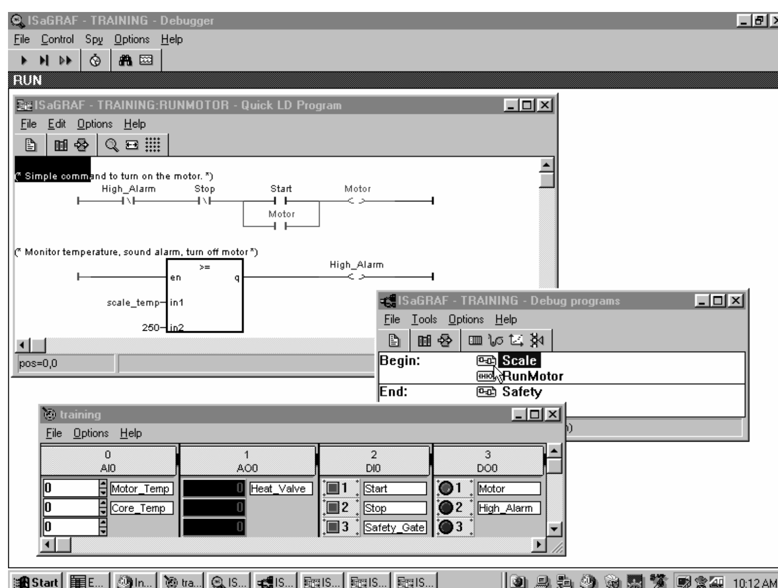
双击 **Scale** 程序。

双击 **Safety** 程序。



然后 →

将这些窗口改变大小和位置以便可以同时观察到。



记住，本应用的**High_Alarm**函数：

如果 **scale_temp >= 200**， 则 **High_Alarm = True** 且 **Motor = False**.

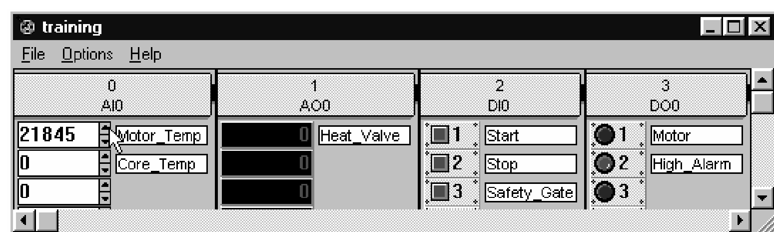
ISaGRAF仿真器不显示内部变量，所以我们只能使用原始温度变量**Motor_Temp**测试。原始温度高限是**21845.**

然后 →

点击 **Motor_Temp** 区。

然后 →

写入 **21840**. 使用 键将**Motor_Temp**值增加至**21845.**



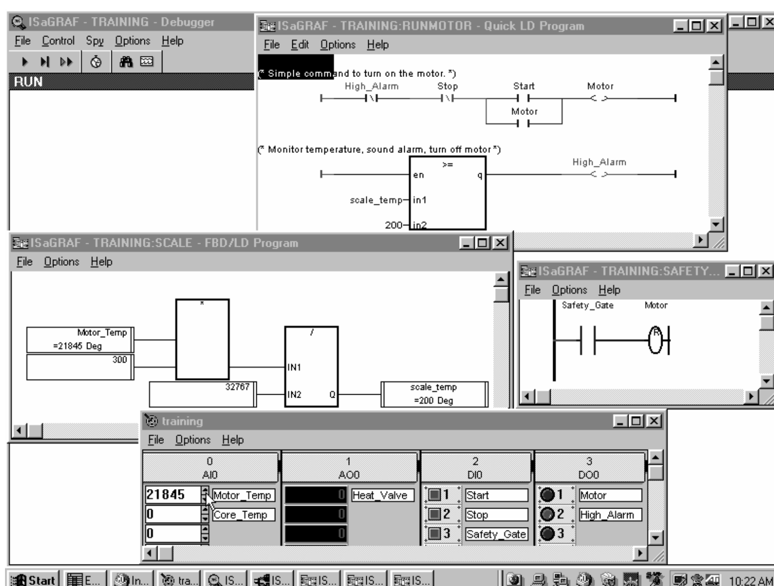
注意当温度值到达 **21845** 时，变量**High_Alarm** LED变亮，而 **Motor** LED 变灭。

然后 →

使用 键将**Motor_Temp**值减少到**21845**以下。则**High_Alarm** LED变灭。

然后 →

右击**Start**按钮，重新启动马达。



然后 →

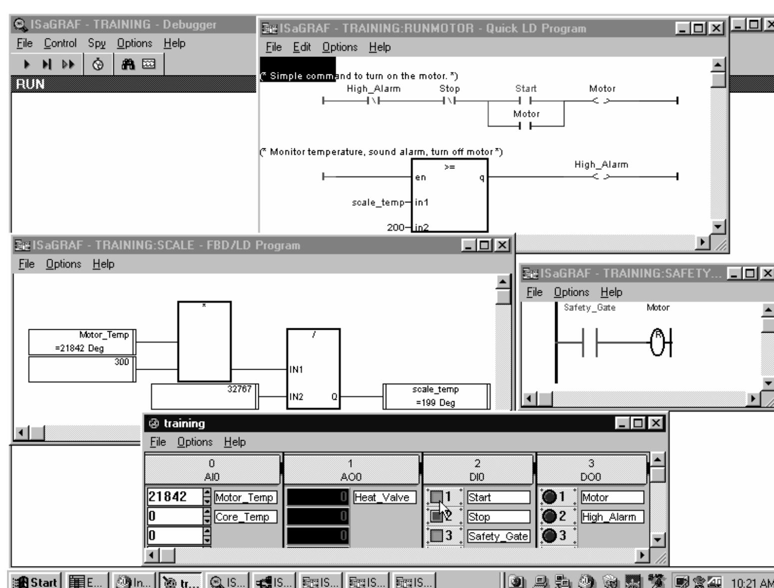
在仿真器上设置 **Safety_Gate** 为 **False**，然后右击 **Start.**

注意 **Motor** 停止运行。

这表明目标循环（target cycle）的重要性。尽管在开始（Beginning）程序中具备启动马达的所有条件， 但Safety程序（End）可以在ISaGRAF写入输出前捕获不安全条件。

然后 →

设置 **Safety_Gate** 为 **True**，马达又可以运行了。



然后 →

从仿真器窗口选择 **File, Exit**，退出仿真方式。

本章我们将要：

- 使用SIXNET Windows Run-time运行您的应用
- 设置 PC/PLC 链接选项
- 使用VersaTRAK I/O仿真器仿真 I/O



关于 ISaRUN -- SIXNET Windows Run-time -- 的更多信息，请参考 SIXTRAK Control Room软件的在线帮助

本章开始前，您应当：

已经完成了本手册的**编译应用**和**使用仿真器**章节， 您的系统必须已经安装SIXNET Control Room软件和SIXNET ISaGRAF Run-time软件。

本例不需要 SIXTRAK硬件， VersaTRAK I/O 仿真器软件将仿真实际 I/O。

IOMAP 和 SIXNET WINDOWS RUN-TIME (ISARUN):

由SIXNET ISaGRAF支持的SIXNET ISaRUN run-time软件是在计算机存储器中的ISaGRAF项目和SIXTRAK I/O数据库或**IOmap**间的接口。

SIXTRAK项目所建立的Iomap是数据库管理器，提供Windows应用和实际I/O的接口。SIXTRAK Iomaps由SIXNET Control Room Iomap工具创建。

Iomap主窗口包含一个用户组态表格，可创建与远程站交换的I/O类型和数量。 Iomap使用这个信息在DLL (动态连接库)设置数据数组合查询远程站的扫描任务。 ISaGRAF和其他 Windows 程序可通过DLL访问Control Room数据库中的I/O。

关于Iomap和 ISaRUN的更多信息请参考 SIXNET在线帮助。

	I/O Type Tag Name	Station Tag Name	Register Count	Update Action
1	Discrete In	SIXTRAK Demo	8	Read Scan
2	Discrete Out	SIXTRAK Demo	8	Write Scan
3	Analog In	SIXTRAK Demo	16	Read Scan
4	Analog Out	SIXTRAK Demo	8	Write Scan
5				
6				
7				
8				
9				
10				
11				
12				

Station Definitions

Window Edit Define Help

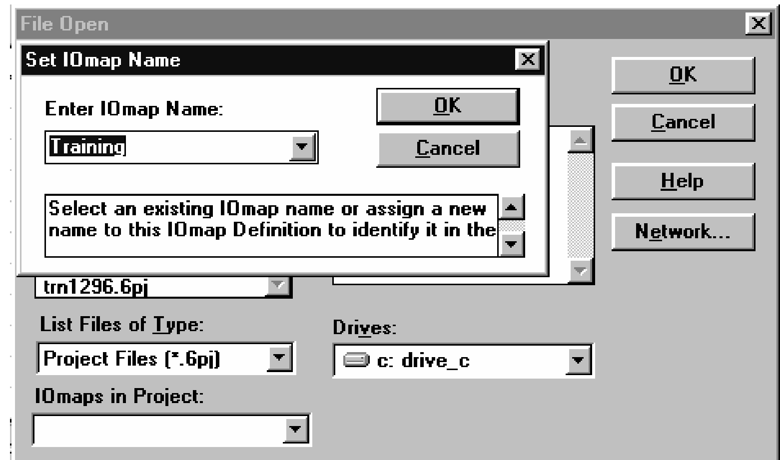
	Station Tag Name	Device	Station Number	Station Type	Status Flag
1	Local Computer	None		IOmap	
2	SIXTRAK Demo	Default	0	SIXTRAK	
3					
4					
5					
6					
7					
8					

Default Device: Ethernet　Close　Help

Allow Duplicate Station Numbers　Auto Load...　Advanced...　Sort...

Each station requires a unique tag name of up to 20 characters. This tag name must match the station's name in other programs. Select Auto Load on the Define menu to load station

开始：



双击 图标，打开 **Control Room IOmap** 软件。

然后 →

选择 **File, New, IOmap.**

然后 →

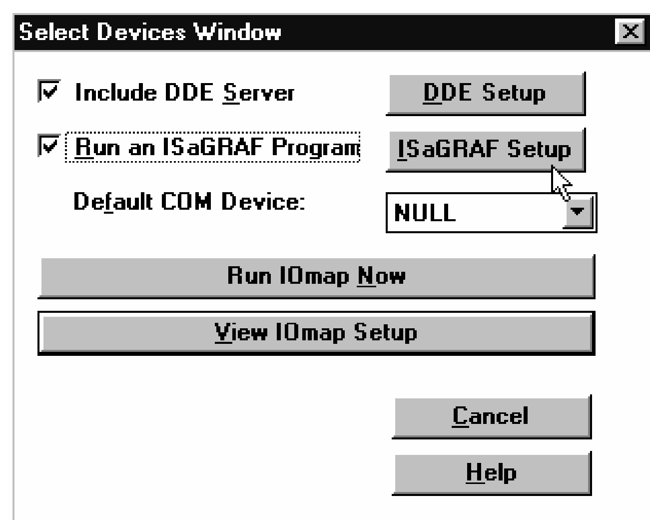
选择 **TRAINING.6pj** 项目文件。

点击 **OK.**

然后 →

将Iomap命名为 **Training**， 点击 **OK.**

注意：在设备窗口，缺省选中DDE服务器。



然后 →

选择 **Yes**，以便在TRAINING.6pj项目使用站定义和I/O寄存器。

Control Room自动执行这个步骤。

使用在线帮助查看选项说明。

然后 →

选中 **Run an ISaGRAF program,** 然后点击 **ISaGRAF Setup.**

然后 →

选择 **Run ISaGRAF.**

在ISaRUN runtime方式运行应用由三种方法：第一种是启动ISaRUN，然后启动ISaGRAF应用；第二种是直接从I/Omap运行已经编译的应用.

然后 →

从Workbench选择 **Start Project.**

选择 **OK.**

然后 →

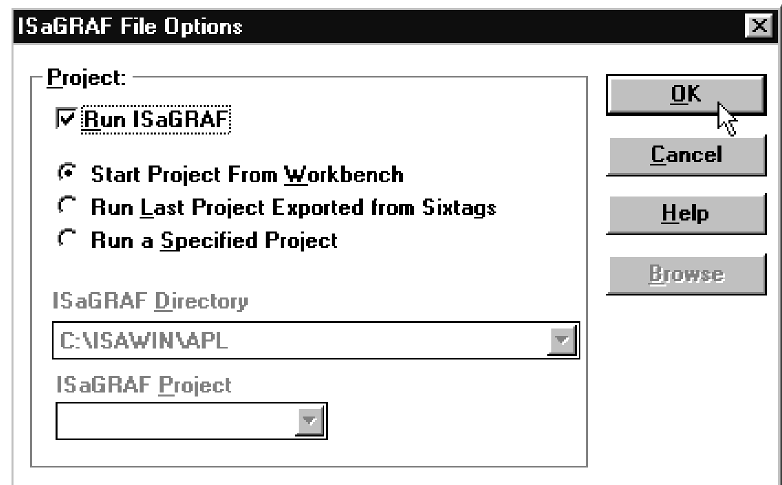
选择 **Null**作为缺省通信设备（Default COM Device）。

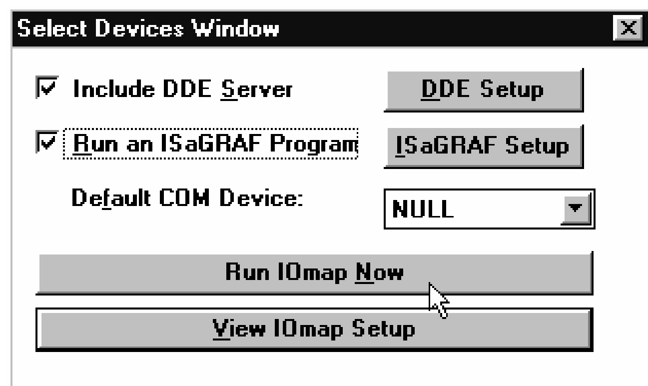
这样，就可以没有外部 I/O，而在计算机上运行了。

然后 →

选择 **Run IOmap Now.**

将会提示您保存组态：选择 **OK.**





ISaRUN runtime将自动启动并最小化，并通知你组态已改变。

然后 →

选择 **OK.**

然后 →

当站定义窗口出现时，选择 **Close.**

选择 **OK**, 然后关闭 IOmap**.**

现在，由三个任务以最小化方式运行：

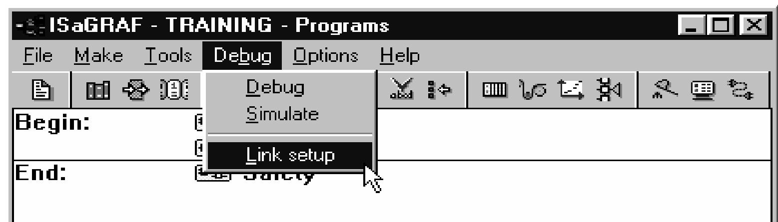
Control Room Scan Task, DDE Server, 和 **ISaRun Windows Runtime.**





然后 →

从**TRAINING**程序管理窗口，选择 **Debug, Link Setup.**

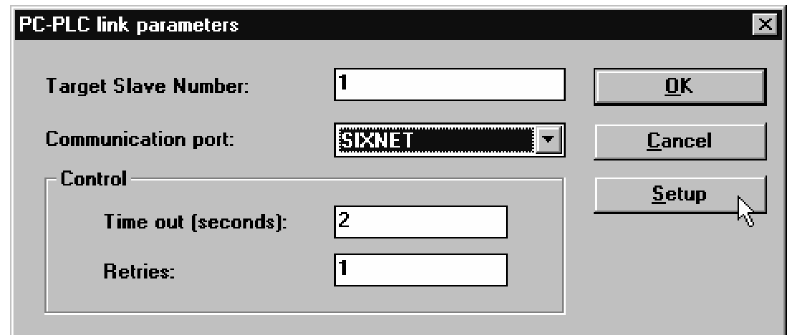


然后 →

在通信口旁的下拉条选择 **SIXNET**.

然后 →

点击 **Setup**.

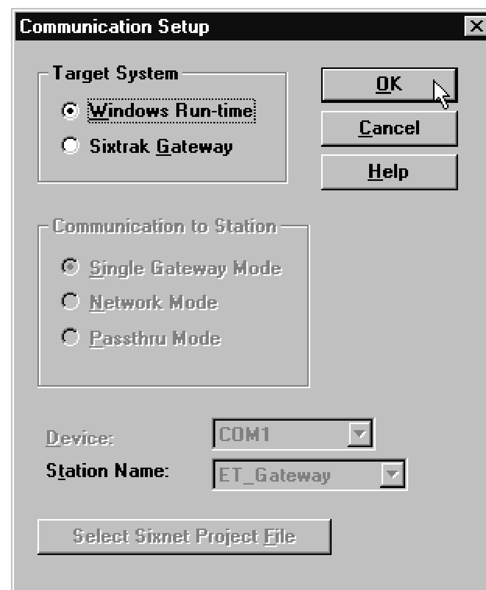


然后 →

选择 **Windows Run-time**.

点击 **OK**.

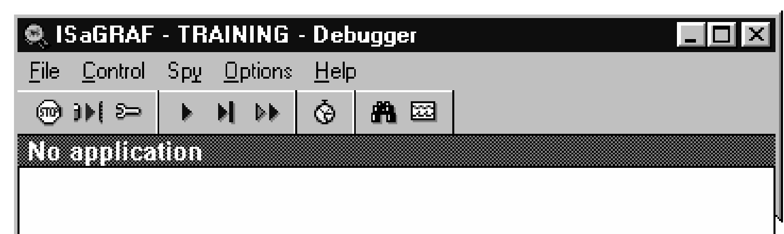
点击 **OK**.



然后 →

从主菜单选择 **DeBug**, **DeBug**.

你将看到调试器窗口出现，但没有应用。



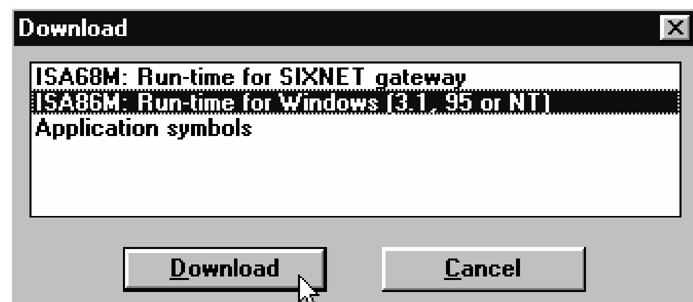
然后 →

选择 **File** ， **Download** 并选择 **ISA86M Windows Runtime.**

然后 →

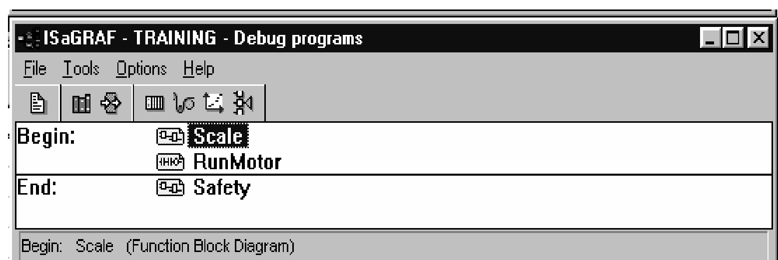
点击 **Download.**

观察下载过程，并注意状态条变为 **Run.**



→

调试器也将程序管理窗口以调试方式打开。本手册也学习调试器的使用。



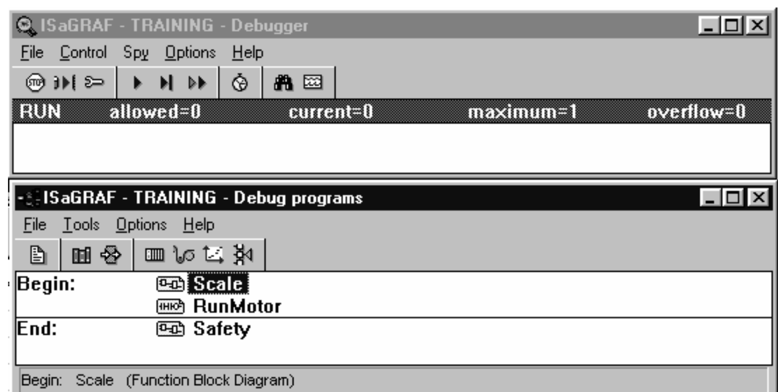
用VERSATRAK I/O将仿真I/O链接到 IOMAP

现在应用在SIXNET ISaRun Windows run-time下运行。可以象在I/O仿真器一样观察应用状态。

在程序管理-调试方式窗口双击图标打开 **RunMotor** 程序。

然后 →

在程序管理-调试方式窗口双击图标打开 **Scale** 程序。



VersaTRAK I/O仿真器是一个调试应用时仿真实际I/O的工具。

然后 →

在SIXNET程序组双击 Simvtrak 图标，

打开 **VersaTRAK I/O** 仿真器。

然后 →

选择按钮 (总在最前面).

然后 →

点击按钮 (设置).



设置屏幕的右边是设置所仿真的项目。

然后 →

选择 **Training.6pj** 项目文件。

选择 **Training** 站。

设置目标I/O为 **Control Room IOmap.**

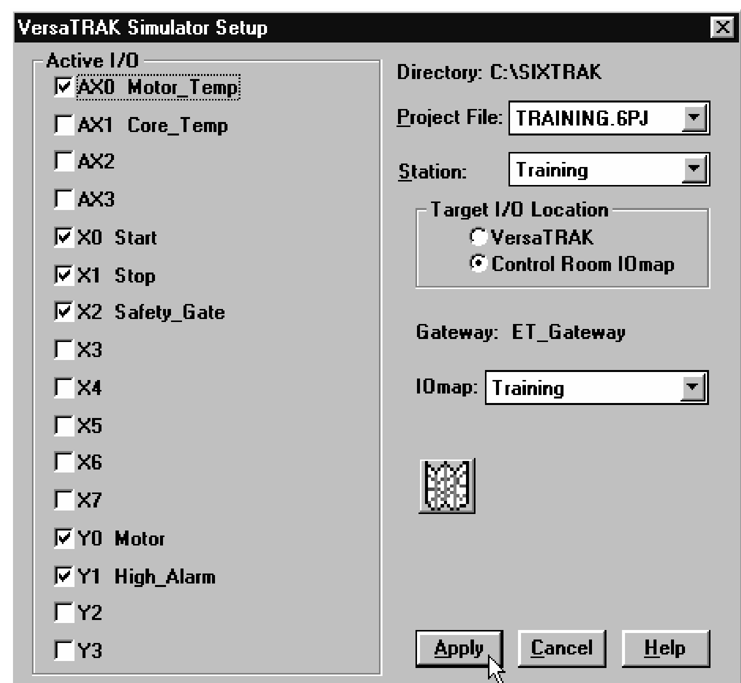
从Iomap下拉菜单选择 **Training** 。

设置屏幕的左边列出所选Iomap中活动的 I/O点

然后 →

选择： **Motor_Temp, Start, Stop, Safety_Gate, Motor, High_Alarm,**

然后点击 **Apply.**



然后 →

将RunMotor和Scale程序窗口和仿真器窗口大小和位置调整，以便可同时看到。

然后 →

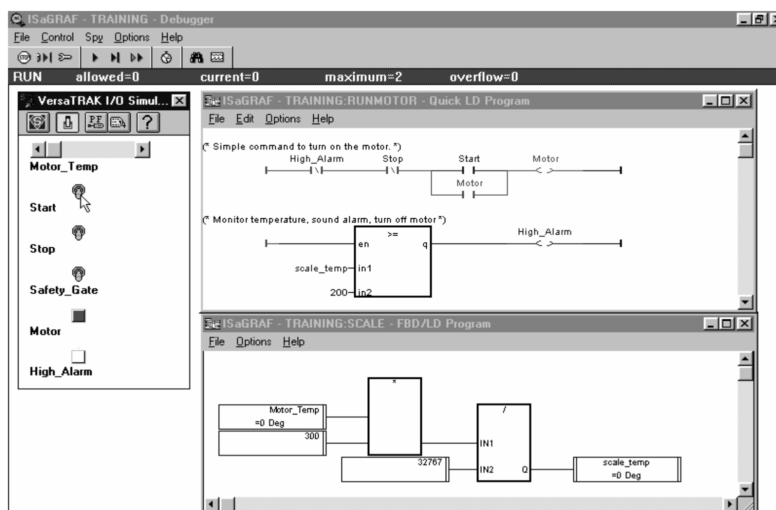
左击 **Safety_Gate** LED，使能马达操作。

然后 →

右击 **Start**按钮启动马达。

注意：现在，仿真其中**Motor LED**变亮。

在RunMotor程序中观察 **Start**接点变为**ON** (由蓝色变为红色).



然后 →

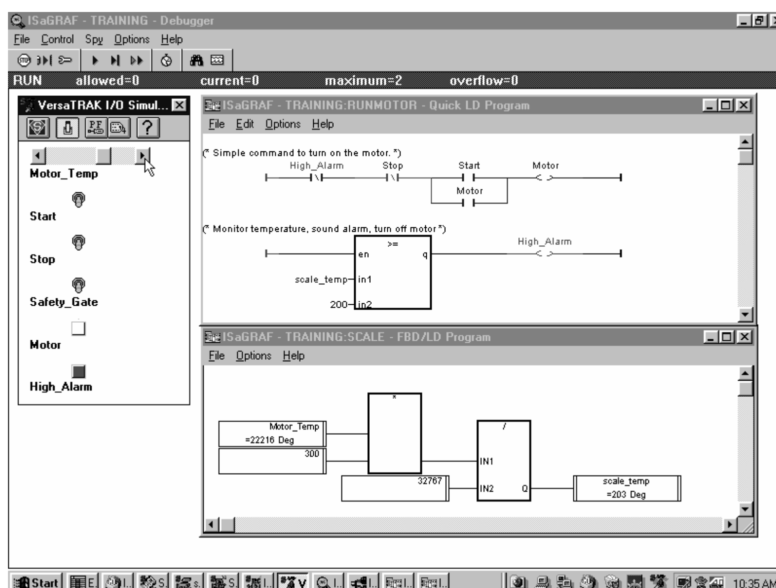
在仿真器中，向右拉**Motor_temp** 滑块。

在**Scale**程序观察相应数值的变化。

注意：当 **scale_temp** 值达到200度以上时， **High_Alarm** 变为 **ON** 而 **Motor** 变为 **OFF**.

然后 →

继续测试您的应用。



然后 →

完成后，关闭 **SIXNET I/O** 仿真器。

然后 →

关闭调试器窗口。

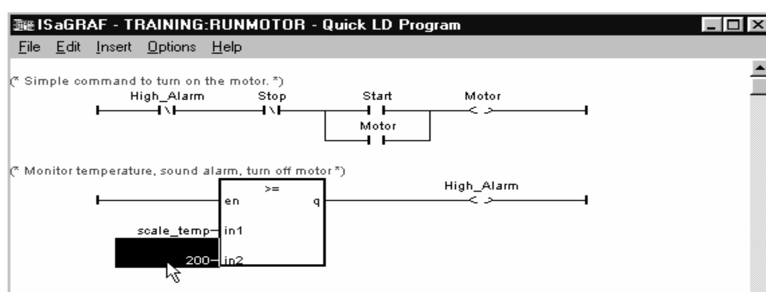
进行程序修改

ISaGRAF应用将继续在ISaRUN run-time下运行。您可以关闭ISaGRAF调试器、Control Room Iomap然后继续编辑ISaGRAF程序。如果改变了程序，应当重新编译并向SIXNET Windows Runtime下载新的版本。

本例假定改变温度由200到**250**时， **High_Alarm** 变为 ON。

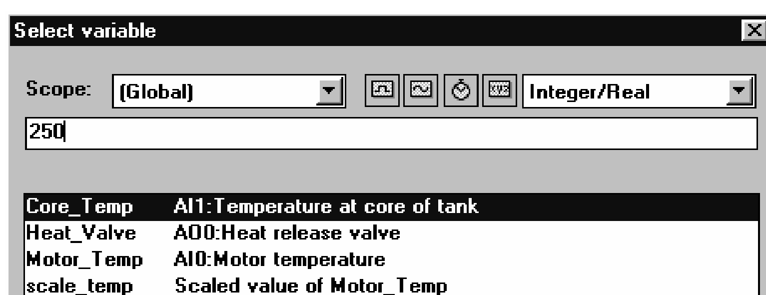
开始：

双击RunMotor 程序图标打开RunMotor程序。



然后 →

双击整型输入200 ，将其改为250.



然后 →

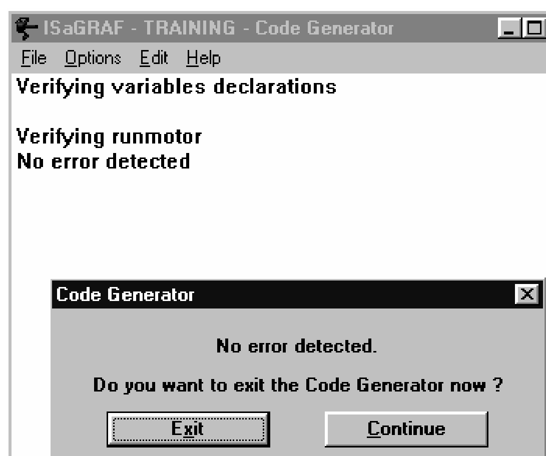
Save , **Verify** 该程序。

然后 →

退出 RunMotor程序。

然后 →

从程序管理窗口选择 **Make, Make Application.**



然后 →

如果编译是没有错误，退出代码生成器。

Tel: 010-62316290 Fax: 010-62316671 support@sixnetio.com.cn

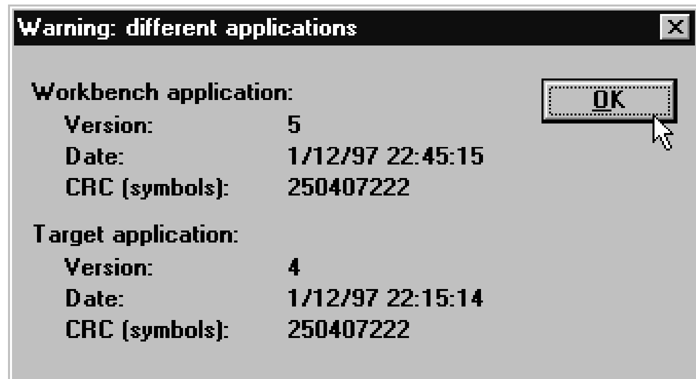
然后 →

从程序管理窗口，选择 **DeBug, DeBug**.

你将得到提示信息：目标应用与当前编译的应用版本不同。

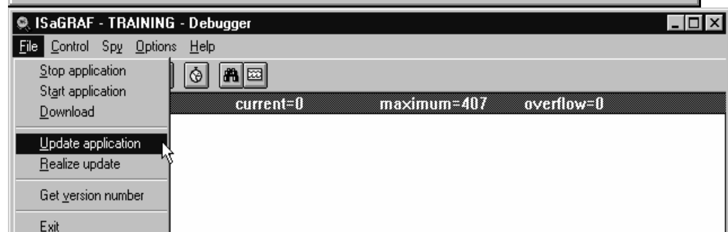
然后 →

点击 **OK**.



NEXT →

从调试器窗口，选择 **File, Update Application.**



然后 →

选择选项 **ISA86M: Run-time for Windows (3.1, 95, NT).**

然后 →

选择 **NOW.**

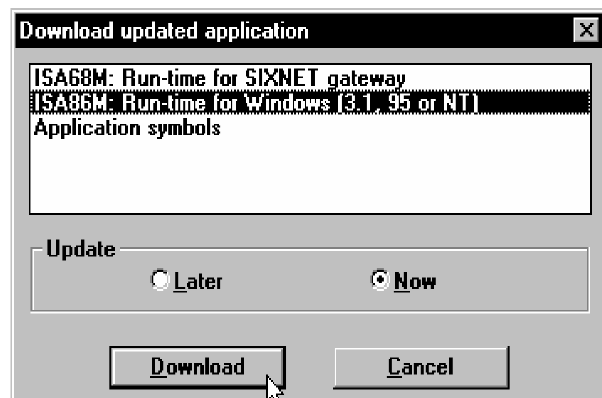
然后 →

选择 **Download.** 您将看到下载新的应用。

按以前的说明打开SIXNET I/O仿真器并确认程序的改变。.

然后 →

完成后，关闭调试器。



关于调试应用的详细信息，请参考使用调试器章。

本章我们将要：

- 使用一个SIXNET可编程控制器运行应用
- 使用VersaTRAK I/O仿真器测试应用
- 设置PC/PLC链接选项



详细信息请参考SIXNET ISaGRAF 帮助文件

开始本章前，您应当：

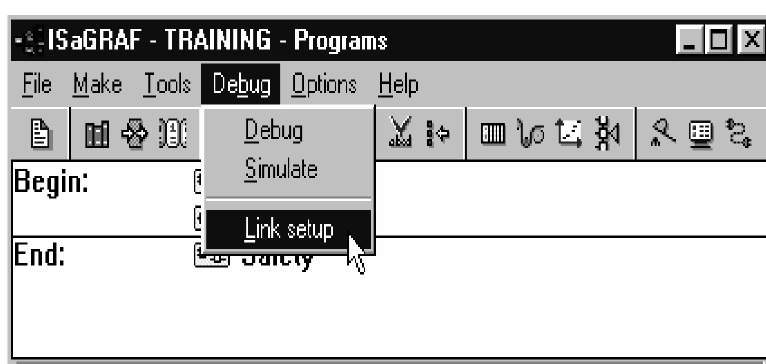
已经完成了ISaGRAF Training项目的创建、编译及使用仿真器章节。

完成本章您需要一个SIXNET或 VersaTRAK可编程控制器，不需要其他I/O硬件。如果您想使用ST-Demo演示单元，只需改变将Virtual I/O改为实际 I/O，并向控制器链接和下载组态。

本例使用以太网控制器(ST-GT-ETH-24P)，通过计算机COM1口连接到控制器的Plant Floor (RS232)口。如果您的硬件设置不同，只需在Plant Floor项目中做很小的修改即可。下列步骤反映了适当的控制器。

开始：

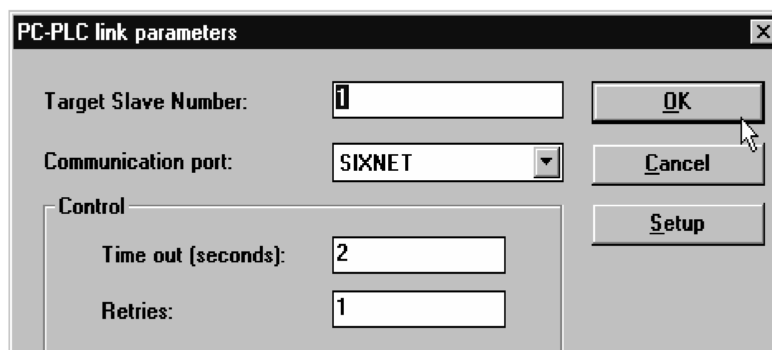
从 **TRAINING**程序管理窗口选择 **Debug, Link Setup.**



然后 →

在通信口选项的下拉框，选择 **SIXNET**.

这个通信口应当总是SIXNET – 不管您是使用 ISaRUN run-time 还是可编程控制器。



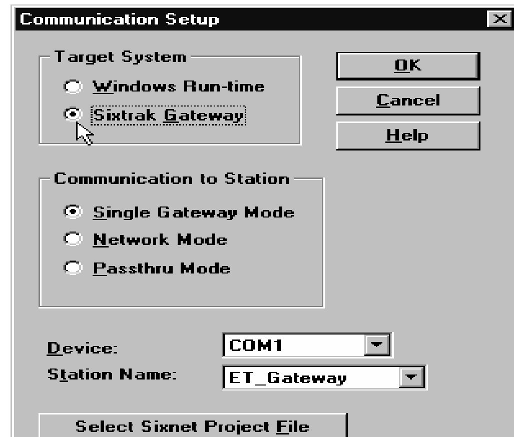
然后 →

点击 **Setup**. 在通信设置的目标系统中选取 **SIXTRAK Programmable Gateway.**

然后 →

点击 **OK**.

点击 **OK.**

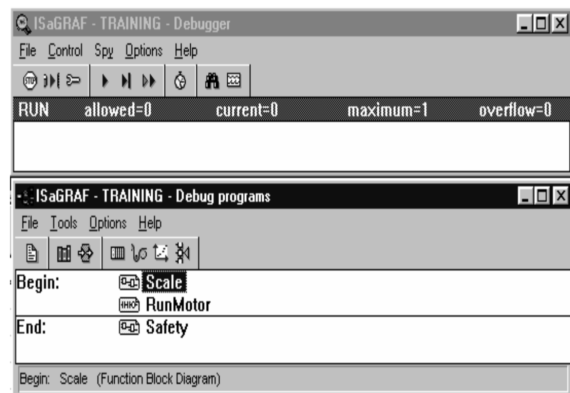


然后 →

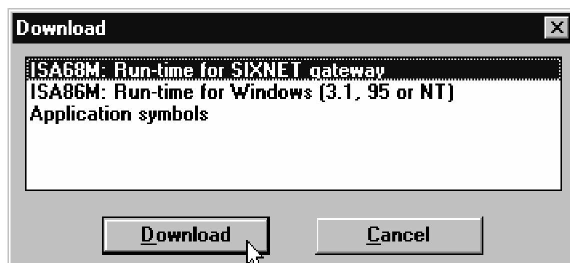
选择 **Debug**, **Debug.**

选择 **File, Download**，并选择 **ISA68M: Run-time for SIXNET gateway**. 将会进行下载，下载进程显示在屏幕上方的状态栏上。

现在，您的应用已经下载到 SIXNET 可编程控制中了。



> *您可使用ISaGRAF调试工具观察应用的运行。参见**使用调试器**章。*

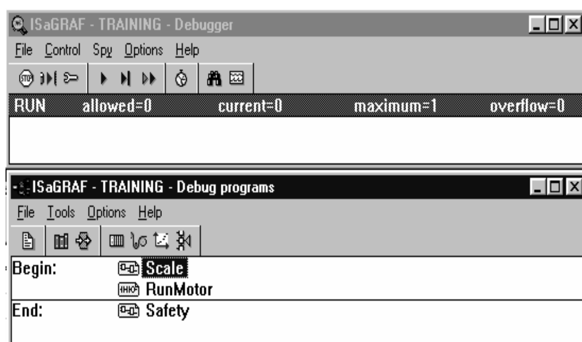


然后 →

在调试方式下，从程序管理器双击程序图标打开RunMotor程序。

然后 →

在调试方式下，从程序管理器双击程序图标打开Scale程序。



VersaTRAK I/O 仿真器工具允许您即使在没有实际I/O情况下，将其发送到SIXNET 控制器。



然后 →

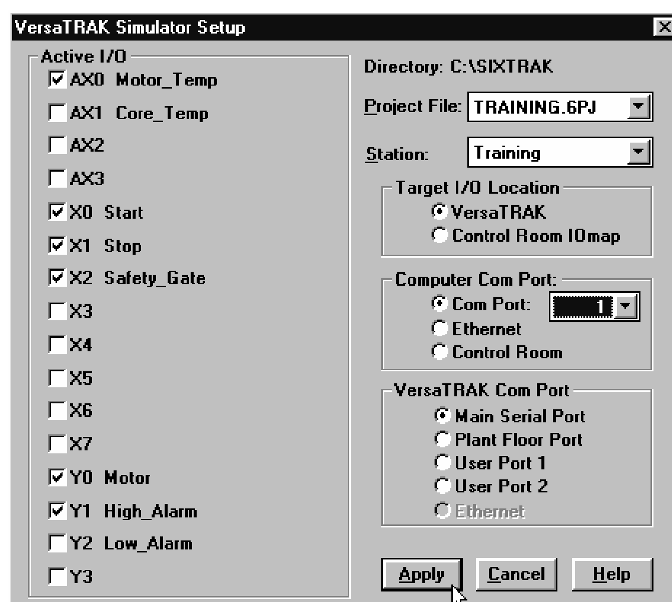
双击 Simvtrak 打开 **VersaTRAK I/O** 仿真器。

然后 →

选择 (总在最前).

点击 (设置) 图标。

设置屏幕的右栏是仿真器使用项目选项。



然后 →

选择 **Training.6pj** 项目文件。

选择 **Training** 站。

设置目标 I/O 位置为 **VersaTRAK.**

选择适当的计算机通信口（如COM 1）。

设置 VersaTRAK 作为主Plant Floor 通信口。

设置屏幕的左栏是该站所有可用I/O点列表。

然后 →

选择: **Motor_Temp, Start, Stop, Safety_Gate, Motor, High_Alarm,**

再选择 **Apply.**

然后 →

调整 RunMotor、Scale程序窗口和仿真器窗口的尺寸和位置，以便可同时观察到。

然后 →

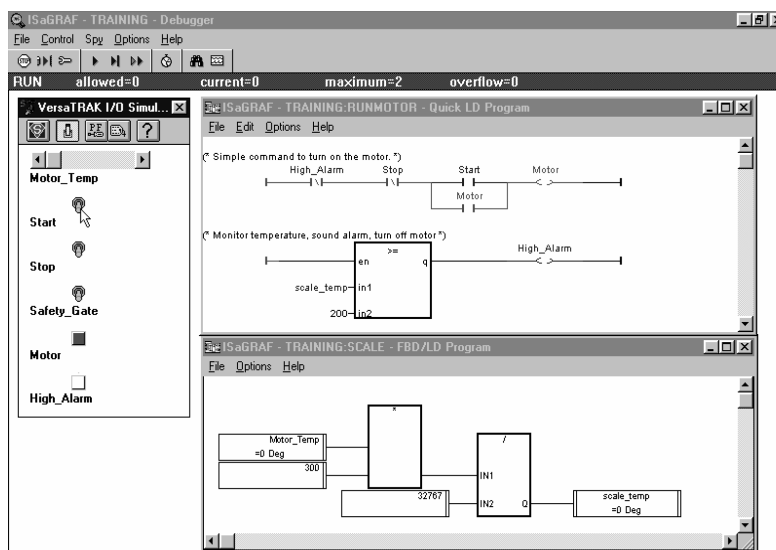
在仿真器窗口左击**Safety_Gate**，使能马达操作。

然后 →

在仿真器窗口右击 **Start** ，启动马达。

注意：此时仿真器中 **Motor LED** 变亮 (**True**)。

如果您有一个ST-Demo演示单元，则 **ST-DO-DCI-08** 的第一个指示灯变亮。



然后 →

在仿真器中，向右拖动 **Motor_temp** 滑条。

在**Scale**程序窗口观察**scale_temp**值，当它升到 **200** 度以上时：

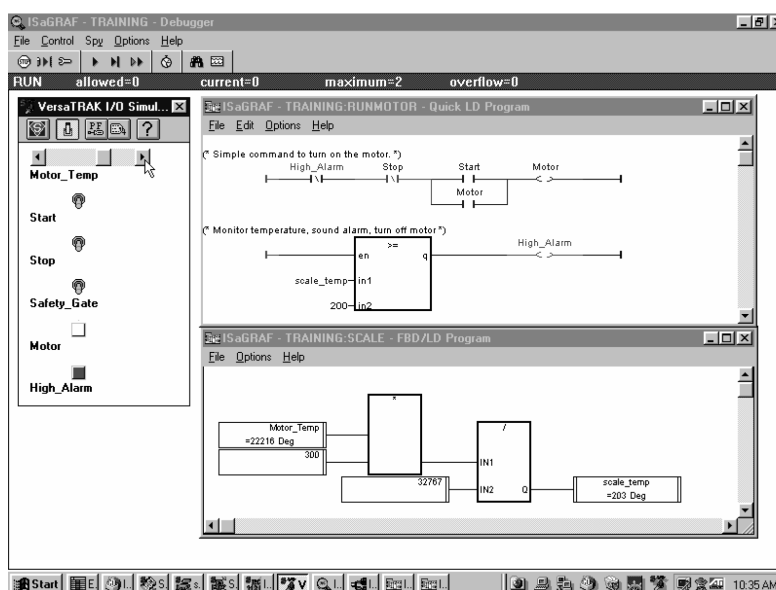
High_Alarm 指示灯变亮。

如果您有ST-Demo演示单元，则**ST-DO-DCI-08** 的第二个指示灯变亮。

马达状态灯变灭。

然后 →

继续测试你的应用。



然后 →

完成后，关闭 SIXNET I/O 仿真器。

然后 →

关闭调试器窗口。

进行程序修改

即使您关闭了ISaGRAF调试器并关闭了计算机，ISaGRAF应用仍然继续在控制器中运行。

如果您继续编辑Training项目，并想实现这个改变，您需要重新编译，并将新版本下载到SIXNET控制器中。在TRAINING程序管理窗口进行这些步骤。

开始：

双击RunMotor图标，打开RunMotor程序。

然后 →

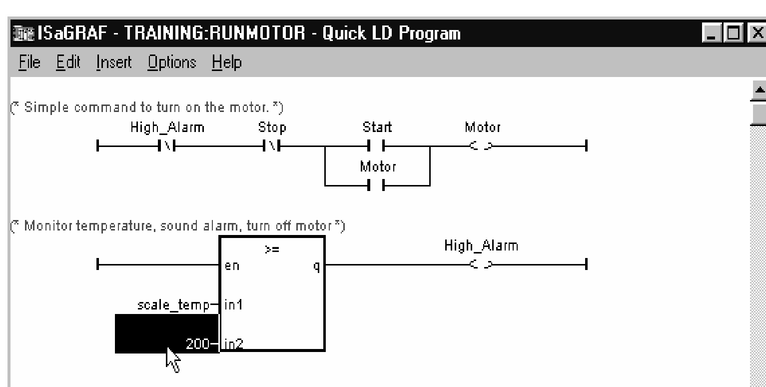
双击整型输入 **200**，将其改为 **250.**

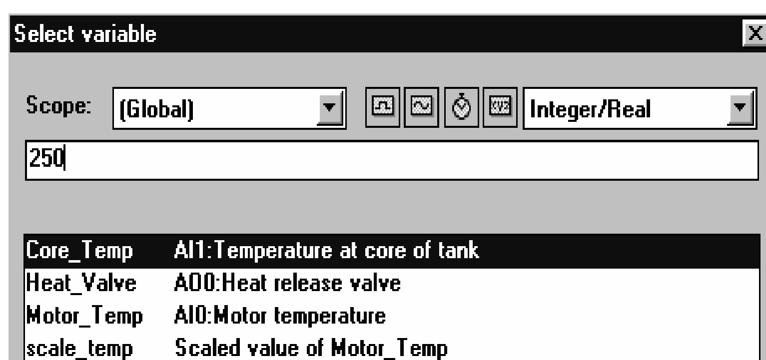
然后 →

保存、确认程序。

然后 →

退出 RunMotor程序。

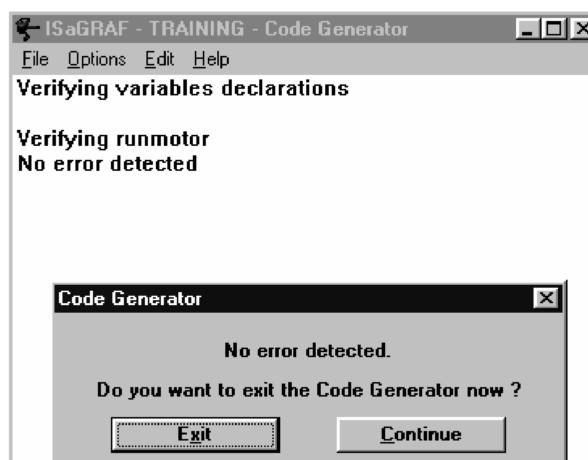




然后 →

从程序管理窗口选择 **Make, Make Application.**

然后退出代码生成器。



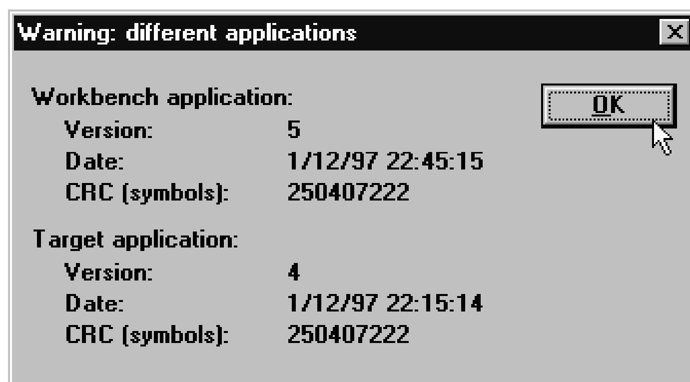
然后 →

从程序管理窗口选择 **DeBug, DeBug**.

您将得到提示信息，说明目标中运行的应用与刚编译的应用版本不同。

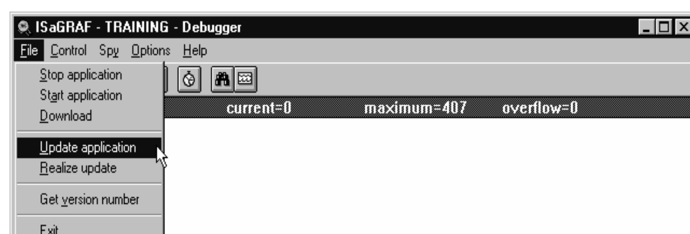
然后 →

点击 **OK**.



然后 →

从调试器窗口选择 **File, Update Application.**



然后 →

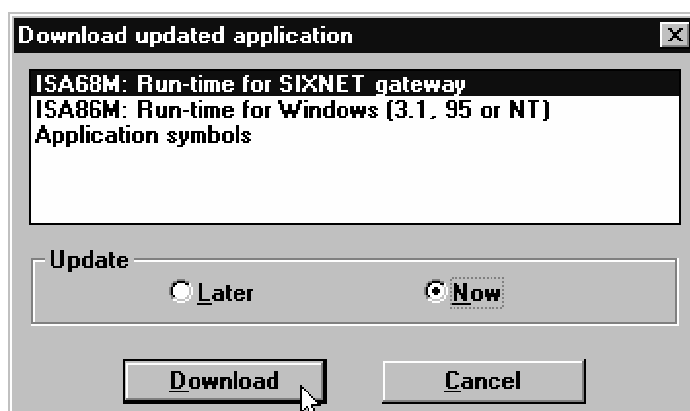
选择选项 **ISA68M: Run-time for SIXNET gateway.**

然后 →

选择立即下载 **- NOW.**

然后 →

选择 **Download.** 您将看到下载过程。



然后 →

使用VersaTRAK I/O仿真器测试改变后的程序。

完成后，关闭调试器和仿真器。

> *关于调试应用，请参考使用调试器章节。*

本章我们将要：

- 回顾调试器控制面板：状态，循环计时，更新
- 使用列表和跟踪变量功能观察I/O数据
- 锁定和写入输入变量



*详细信息请参考ISaGRAF用户手册的**使用在线调试器**章节*

本章开始前，您应当：

已经完成了ISaGRAF TRAINING项目程序的创建和编译，以及本手册揁衪梅抡嫫鳊章。

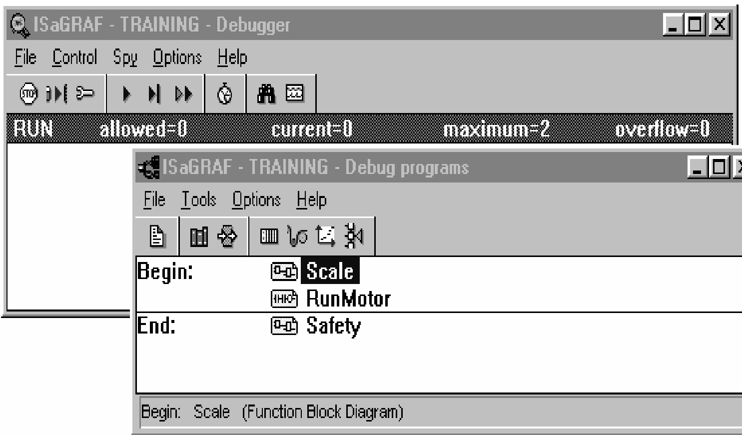
使用图形调试器：

ISaGRAF调试器允许您观察在三种目标系统的控制应用：**ISaRUN Windows run-time**，**SIXTRAK** 控制器, 或 **ISaGRAF** 仿真器。 使用这个强有力的工具，您可以节省大量的开发时间。调试方式下可以查看模拟值，I/O状态变量，扫描时间等。

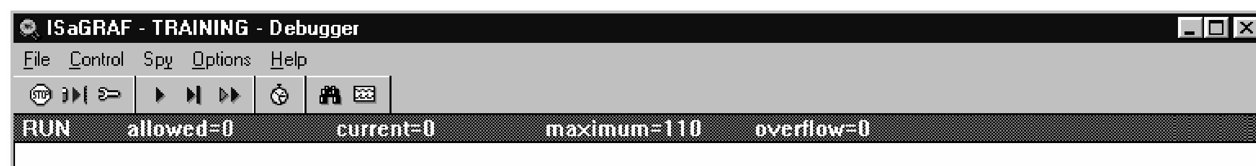


启动时，调试器（debugger）自动以调试方式打开程序管理窗口。调试方式下，可打开ISaGRAF应用的任何程序，但是所有的编辑命令被禁止。

调试方式下，程序元素(步，变迁，变量…)以图形或数值显示当前值，以便容易地跟踪应用的执行。



目标状态和循环定时



这是一个调试器主窗口和控制面板 (在调试菜单)。它显示目标应用的全局状态，和执行循环时间的信息。

可能显示的目标状态：

Logging: 正在与目标系统建立通信

Disconnected: 不能与目标系统建立通信

No Application: 与目标系统的通信正常，但目标中没有ISaGRAF应用

Application active: 与目标系统的通信正常，目标中ISaGRAF应用已激活

Run: 目标应用以实时方式运行

Stop: 目标应用以循环到循环方式（Cycle to Cycle）运行

BreakPoint: 由于遇到断点，目标应用以循环到循环方式运行

Fatal Error: 由于发生严重错误，目标应用故障

显示的循环时间信息：

Allowed: 程序扫描循环计时，0 = 以最大速率运行 (连续)

Current: 上次完整执行循环计时

Maximum: 应用启动后检测到的最大循环

Overflow: 检测到的超过允许时间的执行循环数

所有时间以毫秒计，在仿真方式下不显示时间值。

更多信息请参考ISaGRAF用户手册

关于循环计时：

一般地，程序在目标中以实时方式运行。这意味着执行循环计时由程序条件触发。通过从调试器窗口选择**Control, Cycle to Cycle**，循环可由用户控制一个一个执行 (用**Control, Execute one cycle** 命令)。如果在调试时使用Cycle to Cycle方式，可以使用**Control, Change cycle timing**命令设置允许时间，以毫秒计。

在线修改

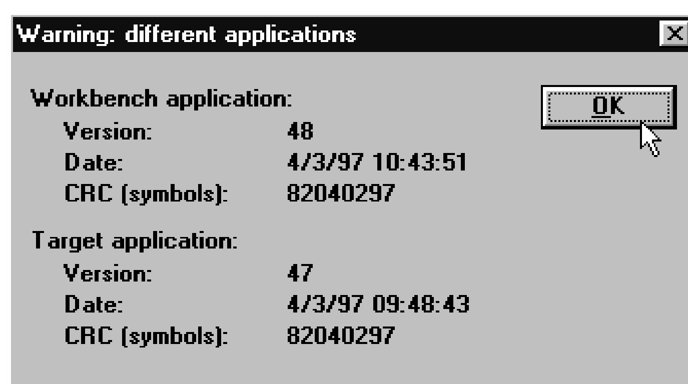
ISaGRAF提供在线修改特性，允许用户在过程运行时修改应用。这在中断过程会危及生产或安全的场

合是需要的。这个功能必须非常小心地使用。

在ISaGRAF程序管理窗口使用**Debug**菜单命令启动调试器。调试器启动时，先确认在目标系统的应用是否与最新编译的项目版本一致。

如果您已经修改应用并发出 **Make** 命令，ISaGRAF将通知您新的版本与目标运行的应用不匹配。

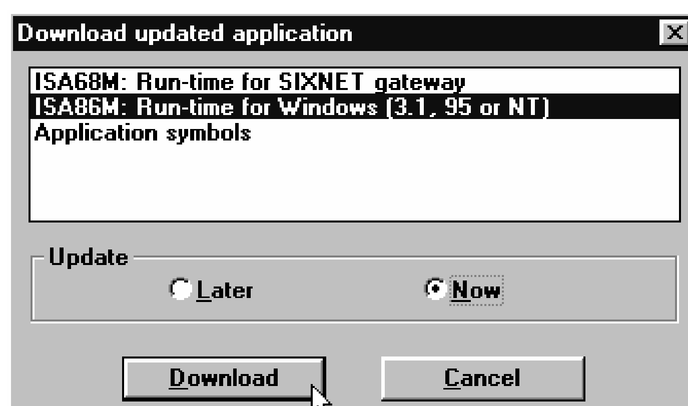
ISaGRAF不自动下载新的版本，请在调试器窗口使用**Download** 或 **Update** 命令下在新的版本。



在线修改时，可从调试器窗口选择**File, Update**命令重新编译和下载。

必须选择目标系统，然后可选择立即更新或以后更新。

系统在线时一些修改不能做，这时，应当停止应用，然后从菜单选择 **File, Download**命令。



关于在线修改的详细信息，请参考 ISaGRAF用户手册

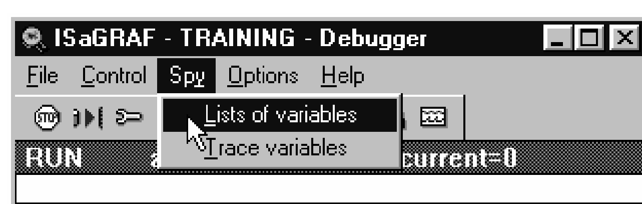
使用监视（SPY）菜单元素：

调试器的监视菜单提供两个便于使用的工具，以便调试时观察 I/O 数据的活动：列出变量（List Variables）和跟踪变量（Trace Variables）。

开始前，应当：

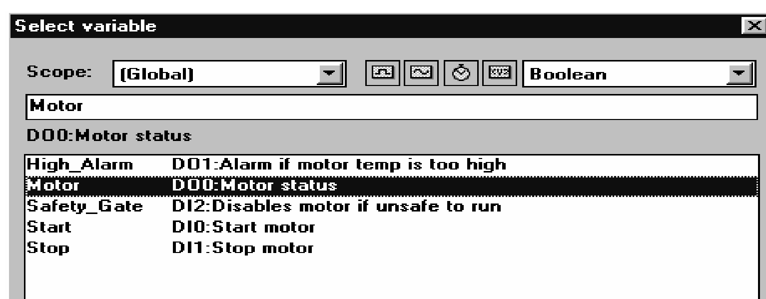
在Windows runtime下运行Training应用，并按照撛赖*indows Runtime运行应用章*舳疸ersaTRAK I/O仿真器。

List Variables是一个在屏幕上观察整型/实型、布尔型或内部变量、计时器、信息的有用方法。一个列表可以放置32个变量。



首先 →

从调试器窗口选择 **Spy, List of Variables.**

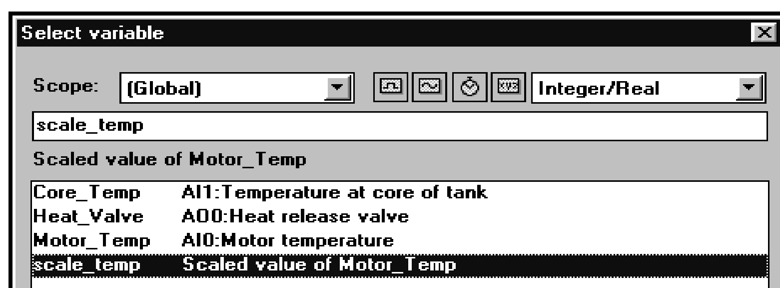


然后 →

点击插入按钮 。

然后 →

使用按钮 显示布尔变量，选择 **Motor.**



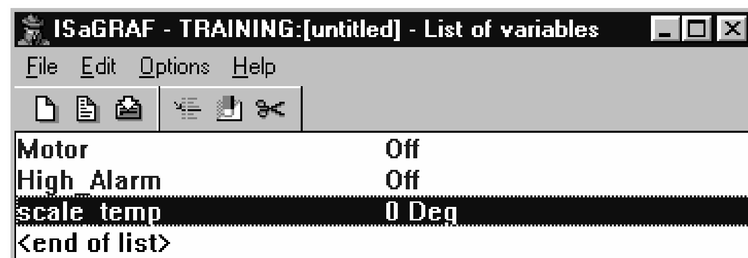
然后 →

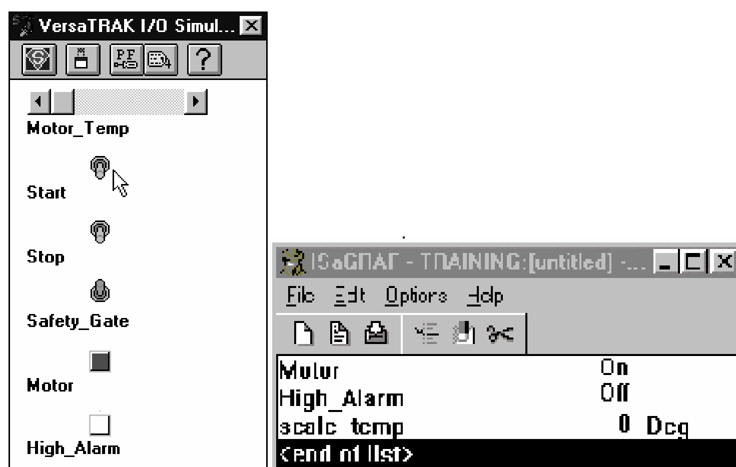
再次点击 选择 **High_Alarm.**

然后 →

点击 ，使用 显示整型/实型变量并选择 **scale_temp.**

变量列表窗口如右图 →





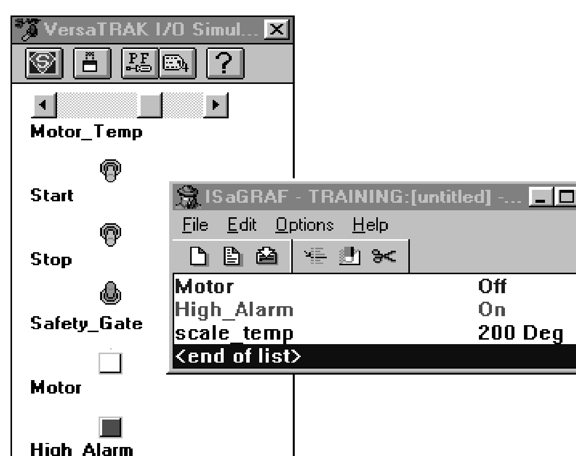
然后 →

在VersaTRAK仿真器上左击**Safety_Gate**使能马达操作。

然后 →

右击**Start**启动马达。

注意：此时变量列表窗口中Motor为**ON.**



然后 →

在仿真器中使用滑动条增加马达温度（原始值）Motor_Temp.

然后 →

在变量列表窗口观察变换后的温度值**scale_temp**。

注意: 当**scale_temp**达到200度时，**High_Alarm**变为**ON** 而 **Motor** 变为 **OFF**。

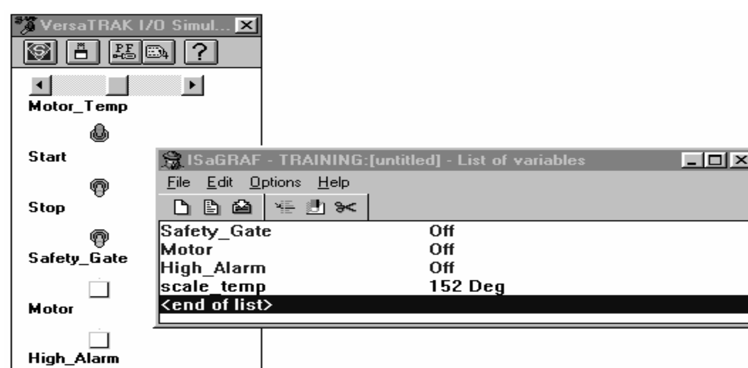


然后 →

在变量列表窗口左击 **Motor** 选择之。

然后 →

点击 按钮选择 **Safety_Gate**.



然后 →

使用VersaTRAK仿真器将**Safety_Gate**变为**OFF.**

然后 →

试着用仿真器启动马达。

由于Safety_Gate是一个故障检测器，您将不能启动马达。

然后 →

在变量列表窗口双击 **Safety_Gate**，选择**Lock.**

这个特性用于在扫描周期锁住输入变量，变量锁住时，ISaGRAF将不写入变量。

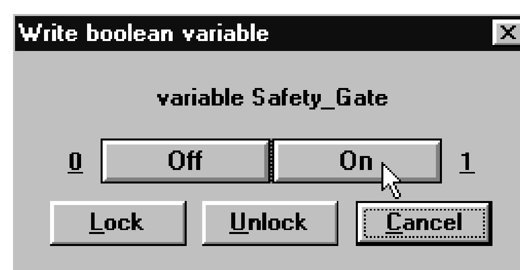


然后 →

在变量列表窗口双击 **Safety_Gate**，选择**ON.**

注意：VersaTRAK仿真器中该值没有变化 – 实际的Safety_Gate寄存器读数为**OFF**.

ISaGRAF现在覆盖实际的I/O寄存器，显示**Safety_Gate**为**ON.**



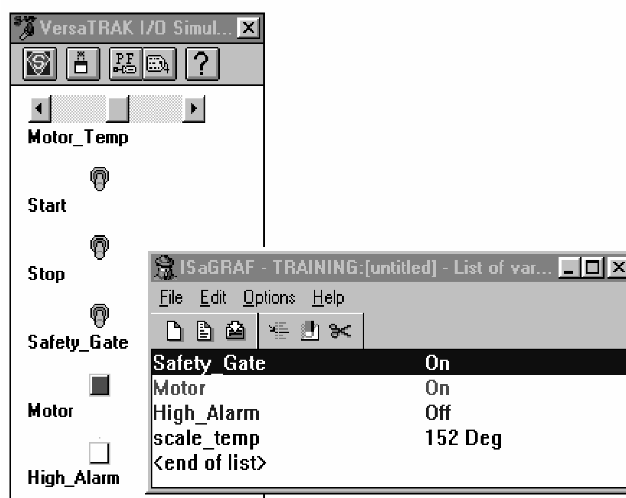
然后 →

在仿真器上右击 **Start**.

在变量列表和仿真器种，**Motor** 都为**ON**.

然后 →

双击 Safety_Gate选择**Unlock**（解锁）**.**



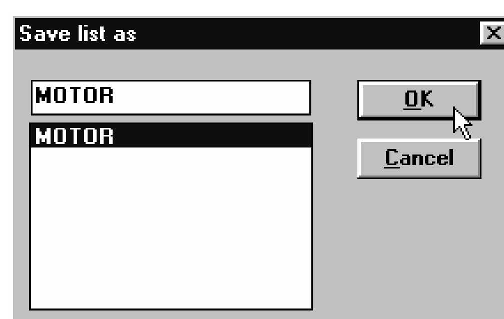
观察已锁住的变量表或对所有变量解锁，请从调试器窗口选择 Control, Unlock all IO Variables。

然后 →

从变量列表窗口选择 **File, Save** 保存变量表。

然后 →

将变量表命名为 **Motor**, 选择 **OK.**



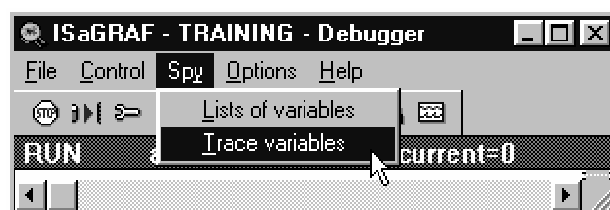
然后 →

退出变量列表窗口.

继续前，应当：

Training应用仍然以Windows runtime方式运行，VersaTRAK I/O仿真器为打开状态。将**Safety_Gate**变为**ON**、将**Motor**变为**OFF**、将**Motor_Temp**复位为**0**.

跟踪变量工具是一个整型/实型、布尔型、或内部变量、计时器或信息类型的实时图形，可以在屏幕上同时观察。同时可观察 **8** 个跟踪变量。

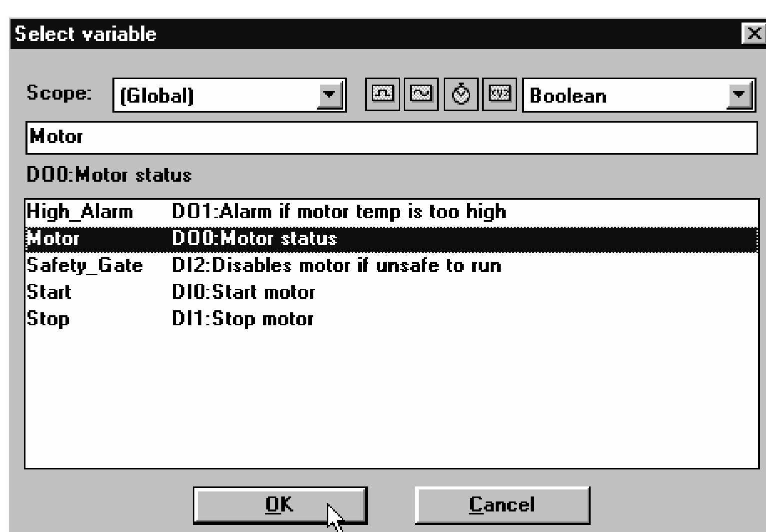


然后 →

从调试器窗口选择 **Spy, Trace Variables.**

然后 →

点击 插入按钮. 您将会看到熟悉的选择变量（Select variable）窗口。



然后 →

使用按钮 显示布尔变量，并选择 **Motor.**

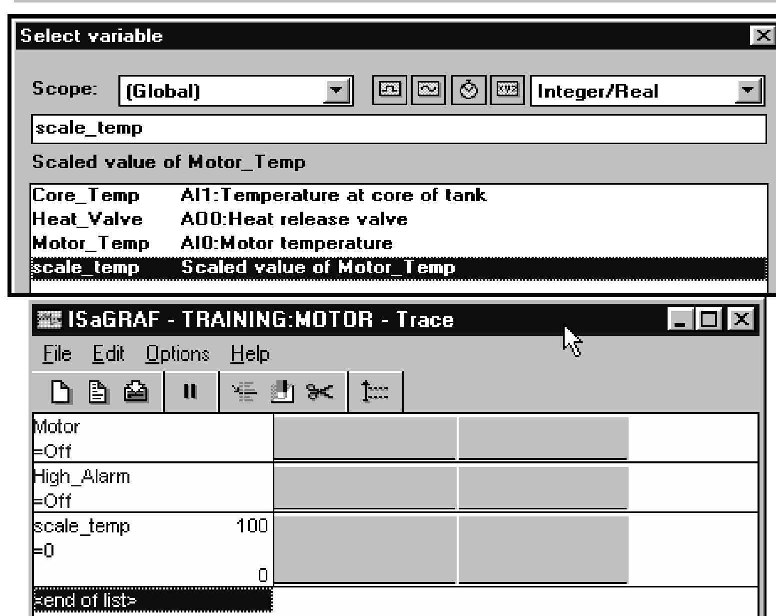
然后 →

再次点击按钮 ，选择变量 **High_Alarm.**

然后 →

点击按钮 ，使用 按钮显示整型/实型变量并选择 **scale_temp.**

变量跟踪窗口如右 →

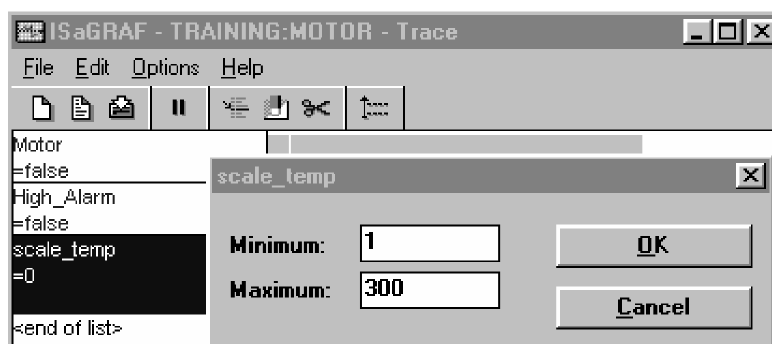


Tel: 010-62316290 Fax: 010-62316671 support@sixnetio.com.cn

然后 →

点击选中变量 **scale_temp**.

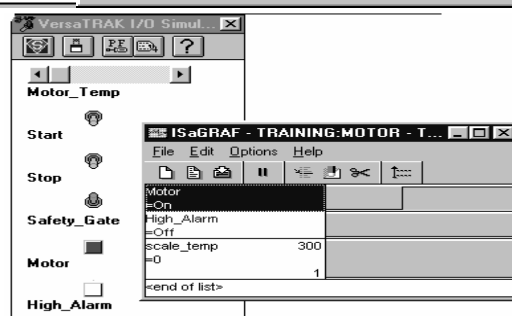
使用设置量程按钮 [button]，选择量程为 **1-300.**



然后 →

使用 VersaTRAK 仿真器启动**Motor.**

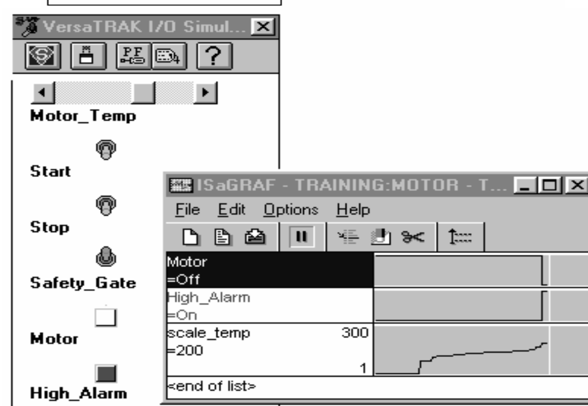
注意: 将Motor 变为**ON** 将立即在跟踪图上反映为一个上升沿。



然后 →

在仿真器重用滑动条增加马达温度值 **Motor_Temp**.

注意: 当 **scale_temp** 达到 **200**时，**Motor** 变为 **OFF**，跟踪图立即显示一个下降沿。

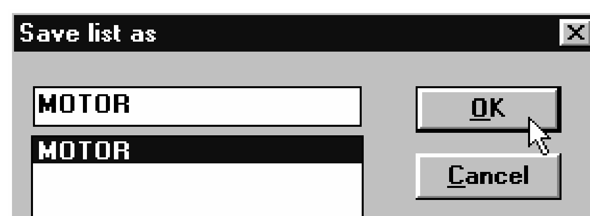


可同上部分一样锁定输入 – 双击变量，选择 Lock，然后重新双击变量，写入要求值。

然后 →

使用 **File, Save** 命令保存这组跟踪变量，将其命名为**Motor.**

退出变量跟踪窗口。



使用调试器的其他工具观察您的应用：

可使用变量字典（**Dictionary**）观察运行应用的变量的当前状态。

首先 →

在调试器窗口点击 ▯ 图标。

在字典屏幕双击变量名可以锁住和写入变量

然后 →

退出字典。

以调试方式打开程序。

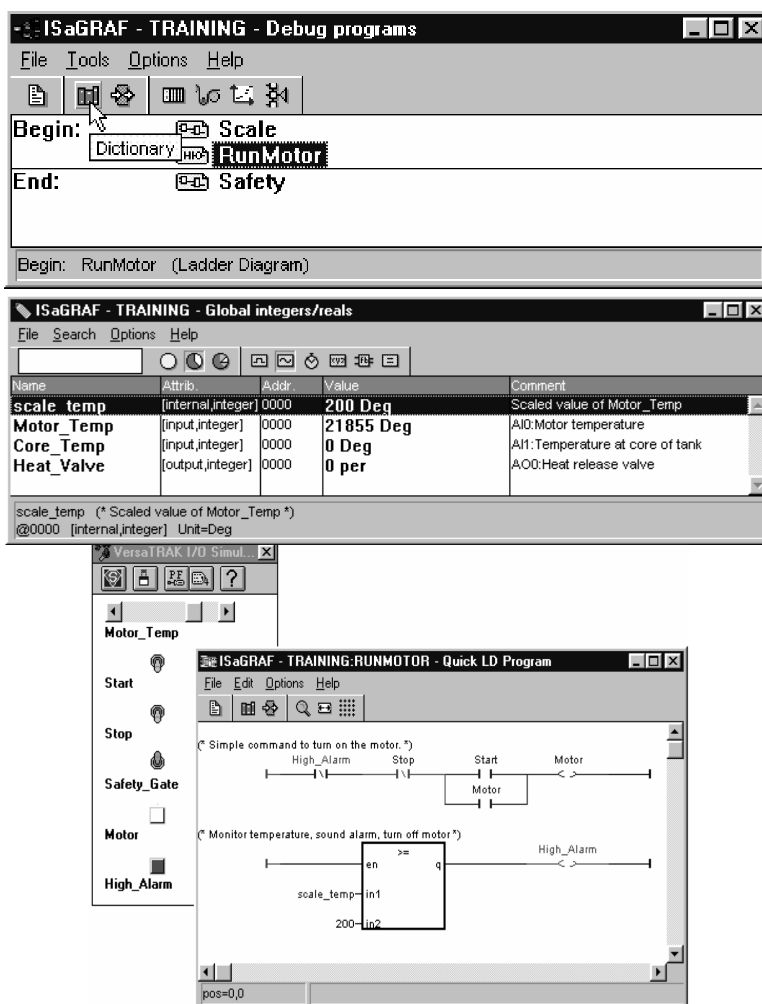
首先 →

从ISaGRAF – Training调试程序屏幕双击**RunMotor**.

然后 →

使用VersaTRAK I/O仿真器启动和停止 **Motor**.

注意: 阶梯和功能块的颜色由蓝色（不通时）变为红色（通时）



在程序中双击变量名可锁住或写入变量

然后 →

退出调试器。

本章我们将要：

- 创建一个ISaGRAF功能块
- 编辑这个功能块
- 在LD/FBD编辑器中的功能块菜单确认之

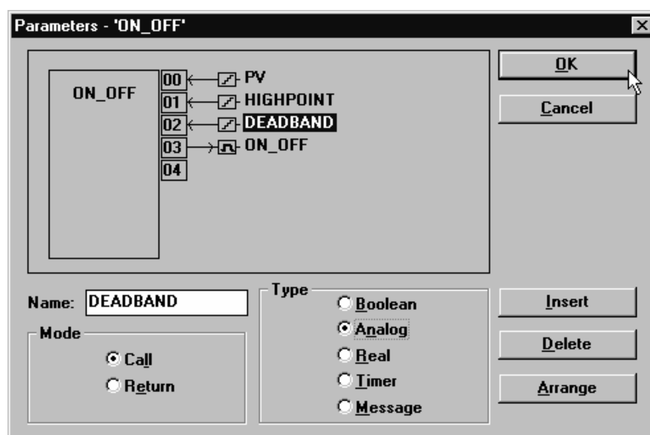
开始本章前，您应当：

已经完成了本手册梯形图和功能块图编辑器的使用章节和I/O仿真器的使用章节。

关于用户自定义功能块

ISaGRAF允许用户使用 **ISaGRAF Libraries**中的工具创建自定义的功能块。可使用标准的ISaGRAF快速梯形图、梯形图/功能块图、结构化文本、或指令表程序编辑器创建*用户自定义功能块，* 不能使用顺序功能表编辑器创建。

功能块的编译和任何ISaGRAF程序或函数一样，使用代码生成器。一个库函数或功能块可以有局部变量和局部定义字。编译后，用户自定义功能块就可在LD/FBD图形编辑器中的功能块窗口被选择了。功能块的归档应使用 **Archive, Function Blocks** 命令。



上图左上方显示块的参数，有调用参数和返回参数。函数和功能块可有多达32个参数(输入或输出)。一个功能块总是有且只有一个返回参数，它必须与该功能块同名。

关于 **ON_OFF** 功能块

我们将创建一个功能块，它接收输入变量 **(PV)** 并判断其值是否在一个范围内，以决定输出的真假。由参数 **HIGHPOINT** 决定范围的高限，而由 **(HIGHPOINT - DEADBAND)** 决定范围的低限。

如果 **PV** > **HIGHPOINT** 则 **ON_OFF = TRUE**

如果 **PV < (HIGHPOINT - DEADBAND)** 则 **ON_OFF = FALSE**

如果 **HIGHPOINT > PV > (HIGHPOINT - DEADBAND)** 则 **ON_OFF =** 当前值

注意：如果您没有完成使用本手册的**LD/FBD 编辑器**章，请先完成。

开始：

双击 图标，打开库工具（**Libraries utility**）

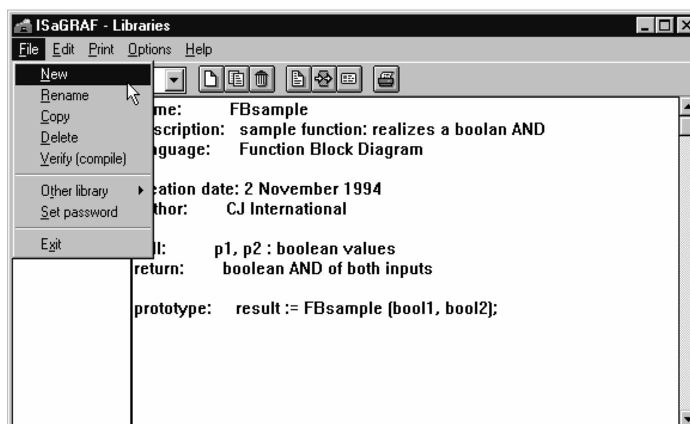


然后 →

在工具栏的下拉菜单选择 **Function Blocks.**

然后 →

选择 **File, New.**

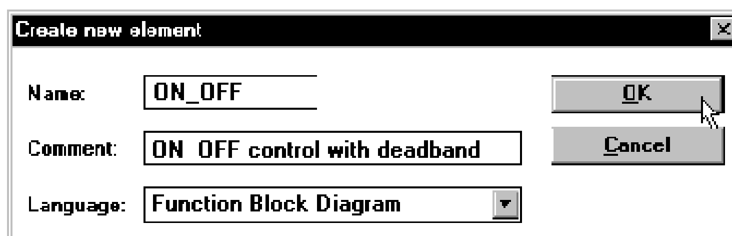


> *功能块的命名：*
> - *不超过 8 个字符*
> - *必须以字母开头，其余部分可以是字母、数字或下划线*
> - *不区分大小写*

然后 →

命名功能块为 **ON_OFF** 并写入注释：

On/Off control with deadband.



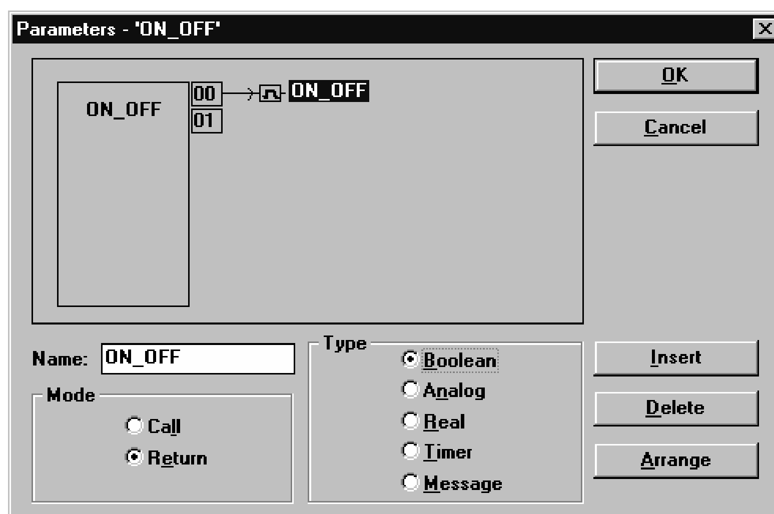
然后 →

选择 **Function Block Diagram** 作为编辑语言。

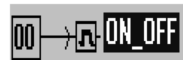
然后 →

选择 **OK**

将出现功能块参数设置窗口。



注意：在块右边的返回参数



，ISaGRAF 自动定义该块的返回参数(输出值)名。

所有的用户自定义功能块返回一个与块名相同的变量。

模式设置为 **Return**.

缺省情况下，ISaGRAF设置类型为模拟，由于 ON_OFF 块要求返回一个真/假值：

将类型设置为布尔型。

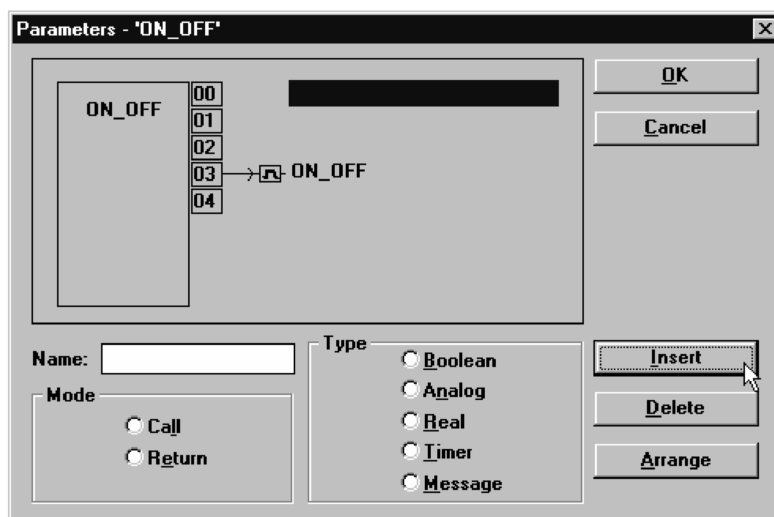
ON_OFF块有三个输入变量（或调用变量）：**PV**， **HIGHPOINT**， 和 **DEADBAND.**

调用变量名与返回变量名：

- *不超过 16 个字符*
- *首字符必须为字母，其余可以是字母、数字或下划线。*
- *不区分大小写。*

然后 →

点击 **Insert** 按钮三次，插入三个新的调用参数。



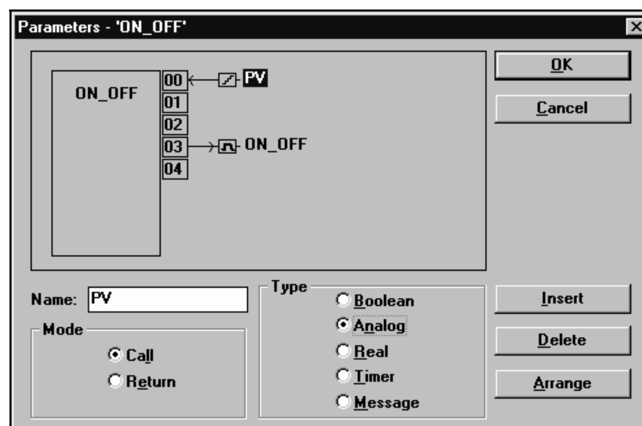
注意：输入将以在表中出现的相同顺序出现在功能块上。

然后 →

点击名字区，向第一个参数写入名字**PV**。

PV是块的输入，将模式选择为 **Call.**

将类型选择为 **Analog.**



然后 →

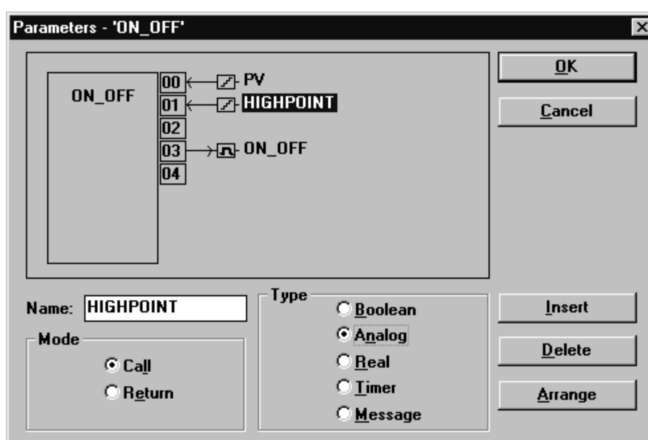
点击输入01旁的小框，选择下一个参数。

然后 →

点击名字区，写入**HIGHPOINT.**

将模式选为 **Call.**

将类型选为 **Analog.**



加上最后一个调用参数：

然后 →

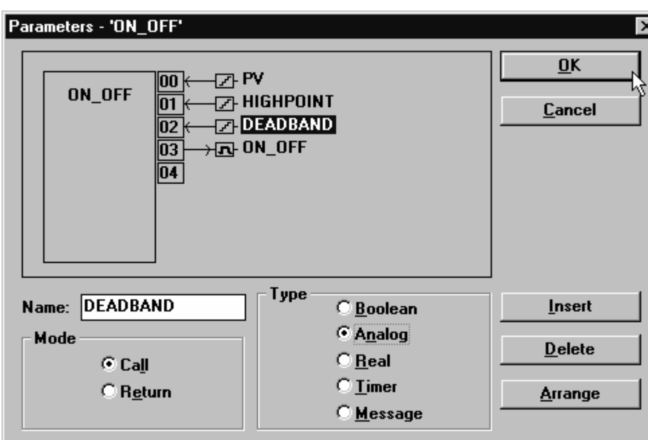
点击输入02旁的小框，选择下一个参数。

在名字区写入**DEADBAND.**

将模式选为 **Call.**

将类型选为 **Analog.**

现在，ON_OFF块的参数如右图所示。



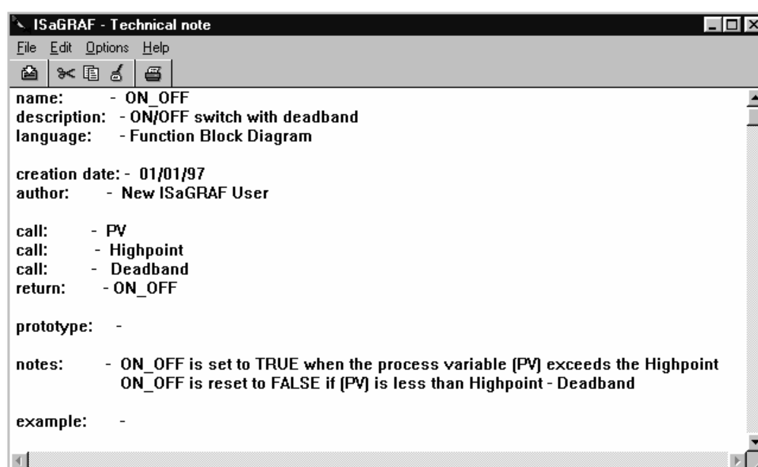
要求变量列表的最后一个是返回参数，点击排列（Arrange）按钮可自动将返回参数排列到最下面。

然后 →

选择 **OK.**

然后 →

从ISaGRAF Libraries库窗口，选择 **Edit, Technical Note.**



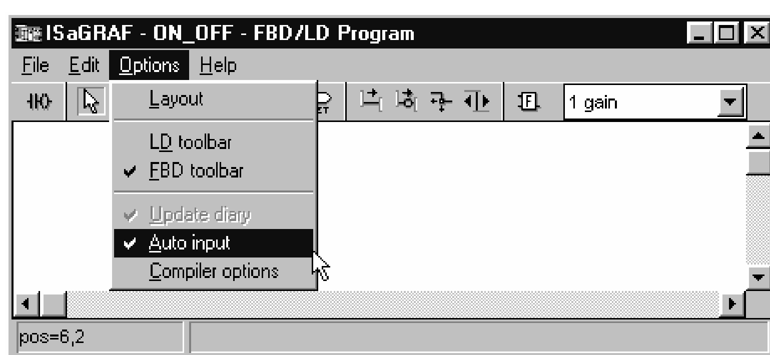
这里，您可以将该块的功能、调用参数、返回参数、作者和其他相关信息整理为文档。

然后 →

如右图所示，写入相关信息，然后保存并退出这个技术笔记编辑器。

然后 →

在ISaGRAF Libraries窗口，双击 **ON_OFF** 块程序，可打开熟悉的LD/FBD编辑器。

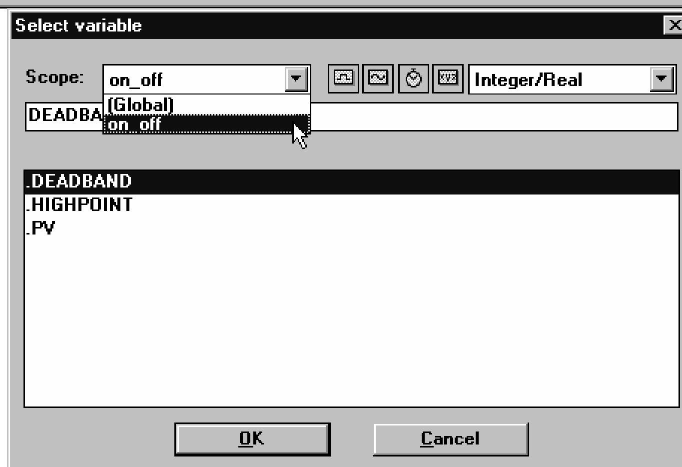


*确保功能块工具条显示出来，并确保选择了选项菜单中的 **verify Auto Input**（检验自动输入）。*

*首先，我们将设置：当**PV**大于**HIGHPOINT**时，**ON_OFF** 为真*

然后 →

选择 按钮，在位置(4,3)放置一个输入变量。



然后 →

出现选择变量窗口时，使用下拉条选择 **ON_OFF**.

点击模拟变量选择按钮，确保模拟变量显示出来。

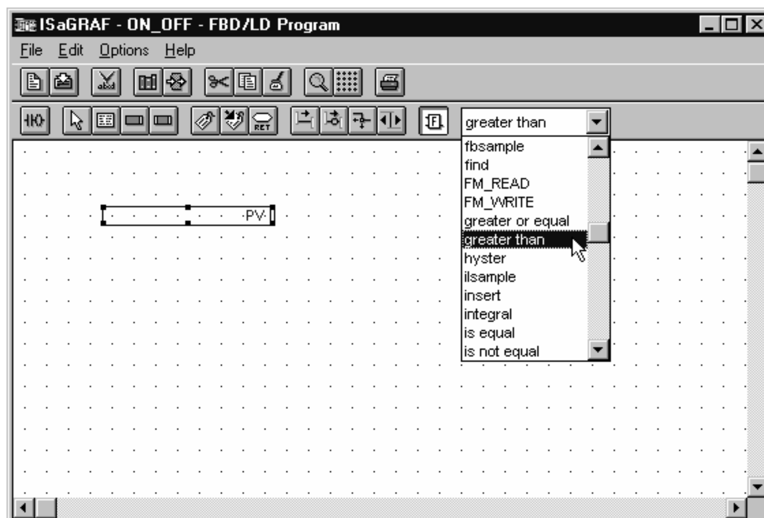
然后 →

选择**PV**，点击 **OK**

然后 →

选择 [F] 按钮，使用下拉菜单选择**greater than** 功能块。

将该块放置到 PV输入的右边，位置(17,2)

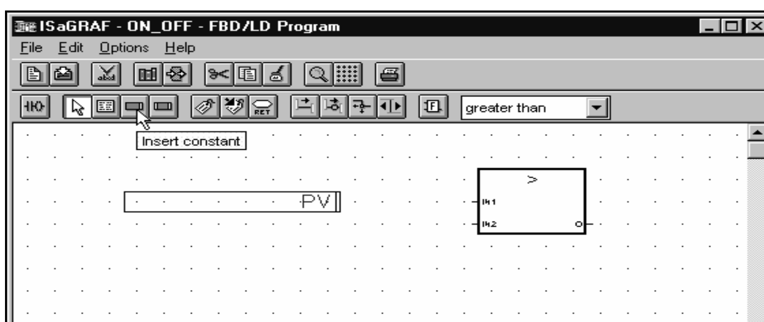


然后 →

选择 按钮，在PV下方放置一个输入变量，位置(4,4)

然后 →

选择变量窗口出现时，选择**HIGHPOINT**，点击 **OK**

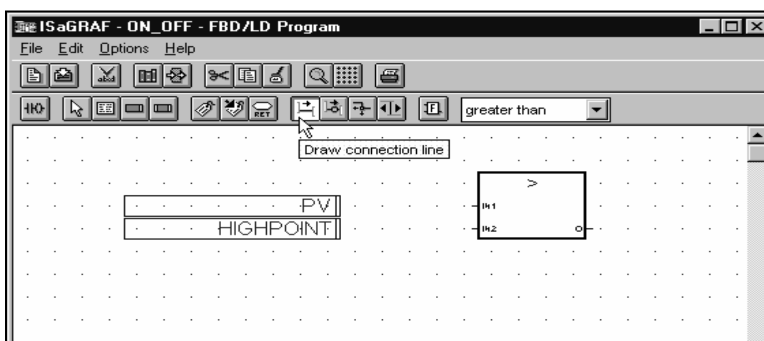


然后 →

选择 按钮，从**PV**到大于块的第一个输入画连接线。

然后 →

从**HIGHPOINT**到大于块的第二个输入画连接线。



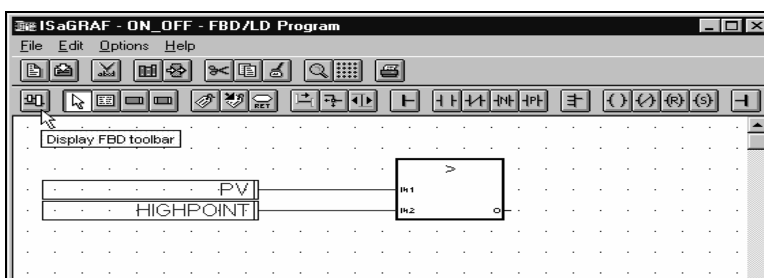
我们将以用一个置位线圈给**ON_OFF**的返回值分配一个**真**值。

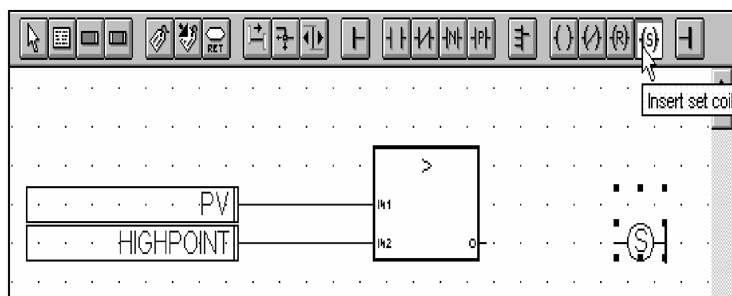
然后 →

使用 触发按钮，弹出**LD**工具条。

然后 →

选择 [set] 线圈，点击位置(26,4)，在大于块的右边放置置位线圈。

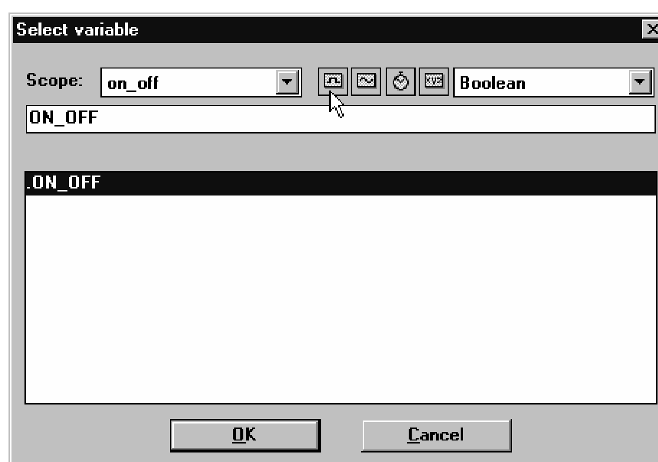




然后 →

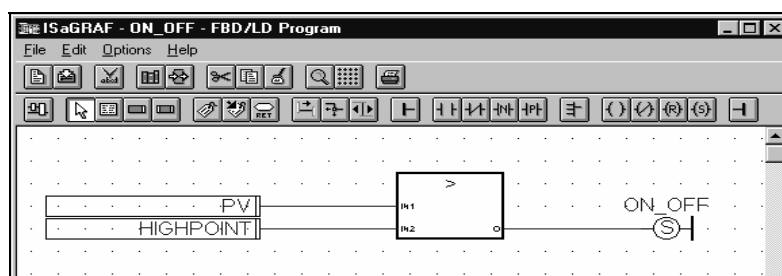
选择变量窗口出现时，点击[Boolean按钮]，显示出布尔变量。

选择**ON_OFF**，点击 **OK**.



然后 →

选择 工具，在大于块的输出与置位线圈之间画连接线。



现在，我们将设置：当 PV 小于 HIGHPOINT-DEADBAND 时，设置 ON_OFF 为假。

然后 →

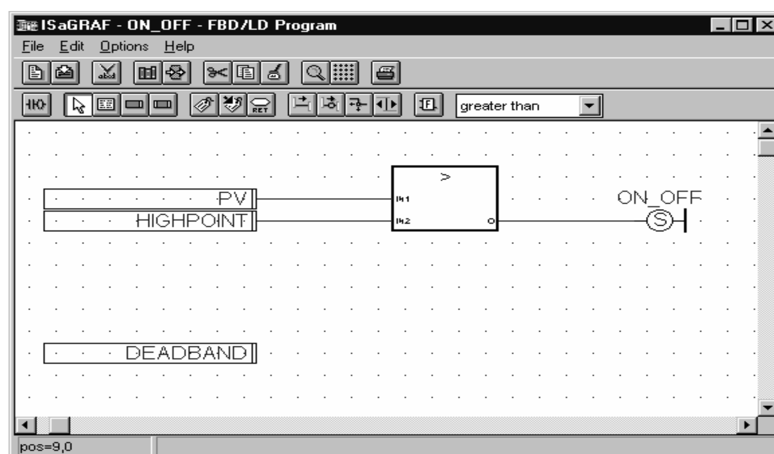
使用 触发按钮，弹出FBD工具条。

然后 →

选择 按钮，在位置(4,10)放置输入变量 **HIGHPOINT**。

然后 →

选择变量窗口出现时，点击[Analog



按钮]，显示出模拟变量。

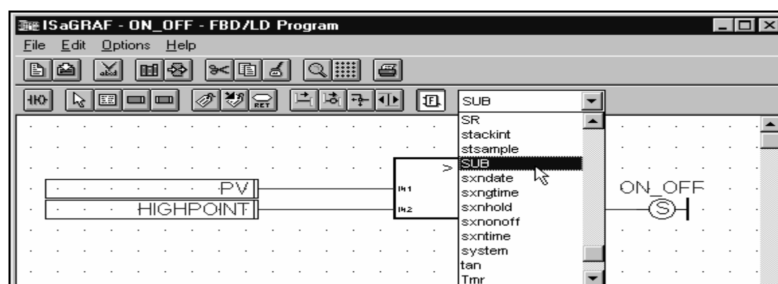
选择 **DEADBAND.**

选择 **OK**

然后 →

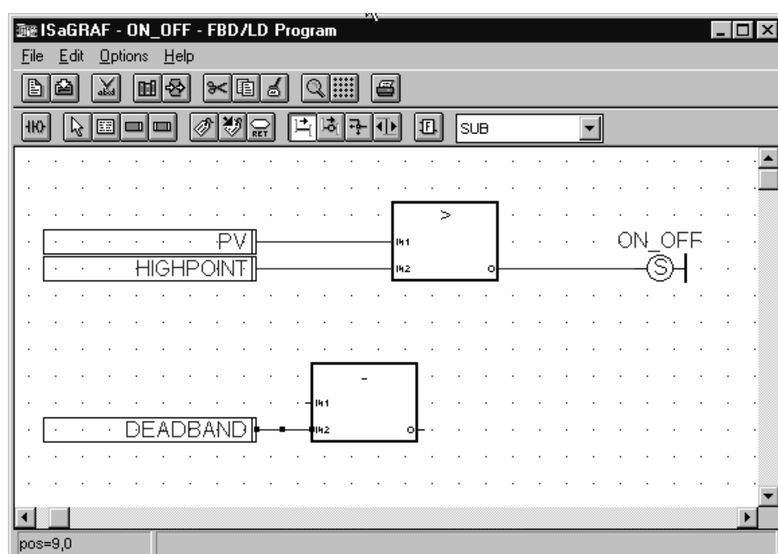
从工具条选择 按钮，使用下拉菜单选择 **SUB**(减法) 功能块。

将该块放置到PV右侧，位置(14,8)



然后 →

选择 按钮，在**DEADBAND**和减法块的第二个输入间画连接线。

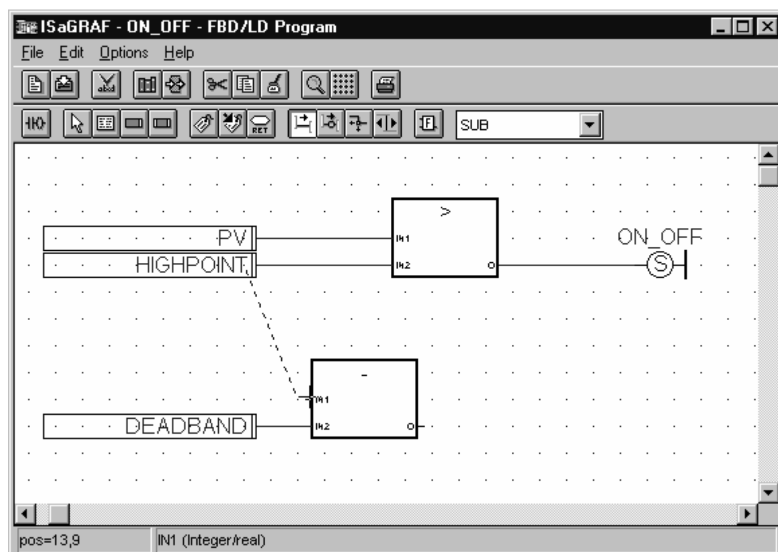


ISaGRAF 允许使用上面的输入变量HIGHPOINT作为减法块的输入。

然后 →

在 **HIGHPOINT** 和减法块的第一个输入间画连接线。

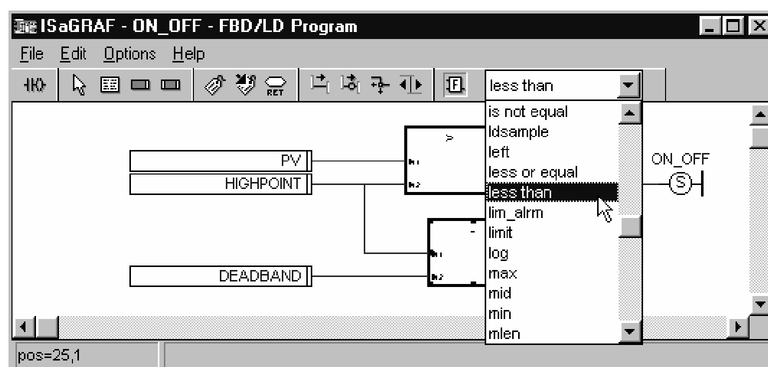
ISaGRAF自动为新块创建一个分支。



然后 →

从工具条选择 [F] 按钮，使用下拉菜单选择 **less than** (小于) 功能块。

将该块放置到 PV 输入的右边，位置(20,6)

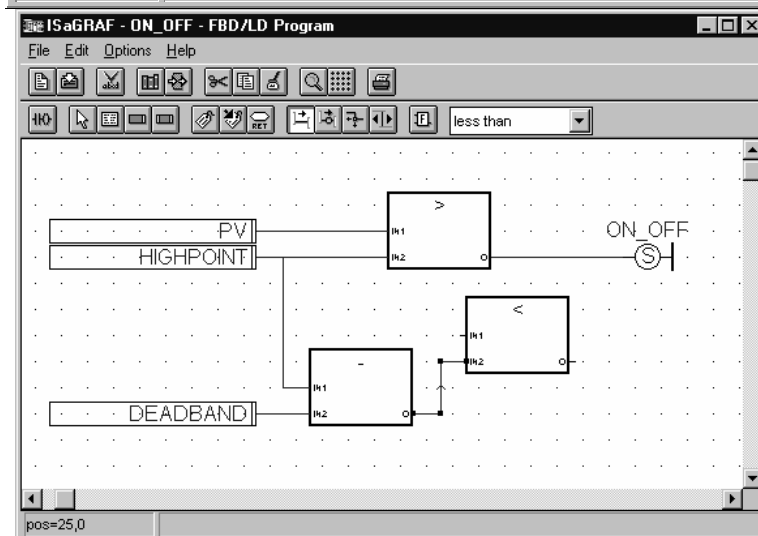


然后 →

选择按钮 ，在 **SUB** 块的输出与 **<** 块的第二个输入间画连接线。

然后 →

在**PV**与 **<** 块的第一个输入间画连接线。



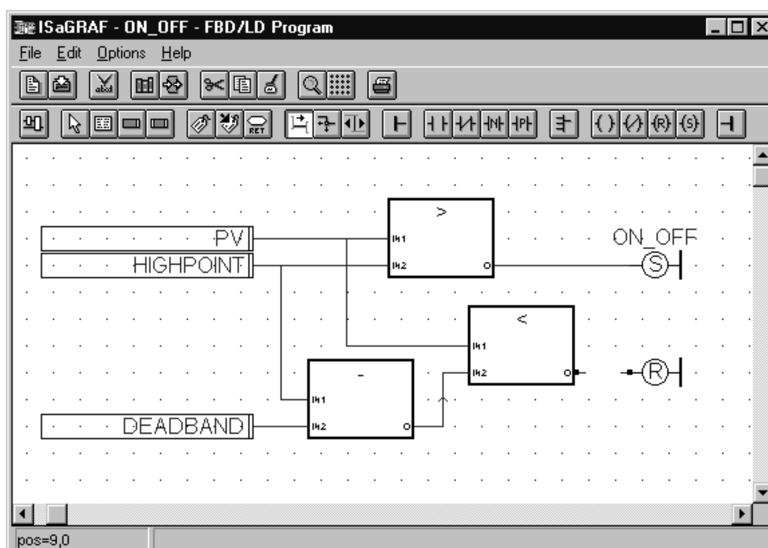
我们将使用一个复位（ reset）线圈给**ON_OFF**的返回参数分配一个假值。

然后 →

使用 触发按钮，弹出 LD 工具条。

然后 →

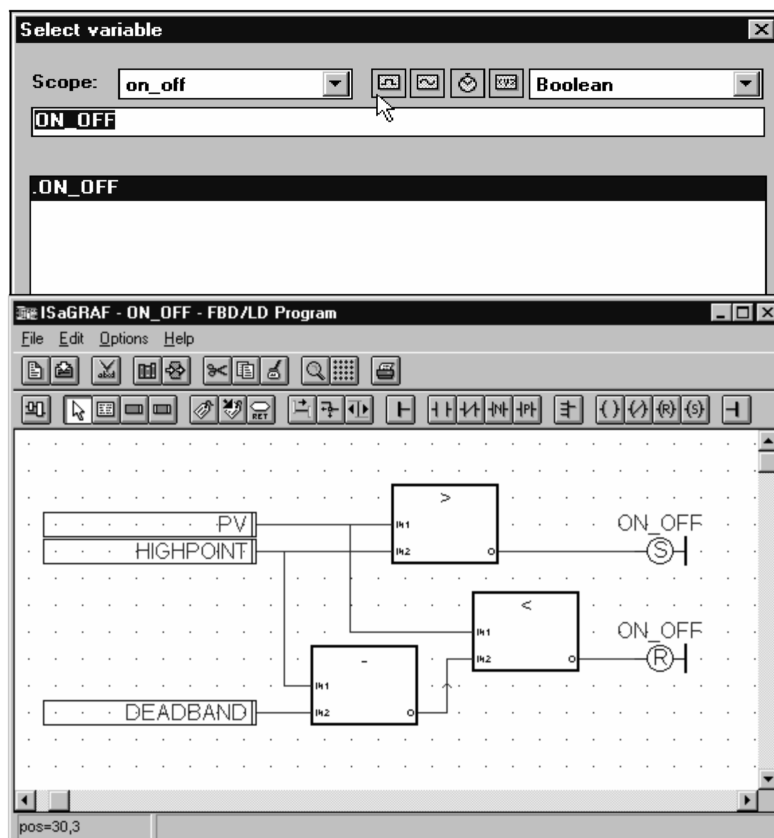
选择 [reset] 工具， 点击位置(26,8)，在小于块的右侧放置一个复位线圈。



然后 →

选择变量窗口出现时，点击[Boolean 按钮]，显示出布尔变量。

选择 **ON_OFF** 并点击 **OK**.



然后 →

选择按钮 ，在小于块的输出和复位线圈之间画连接线。

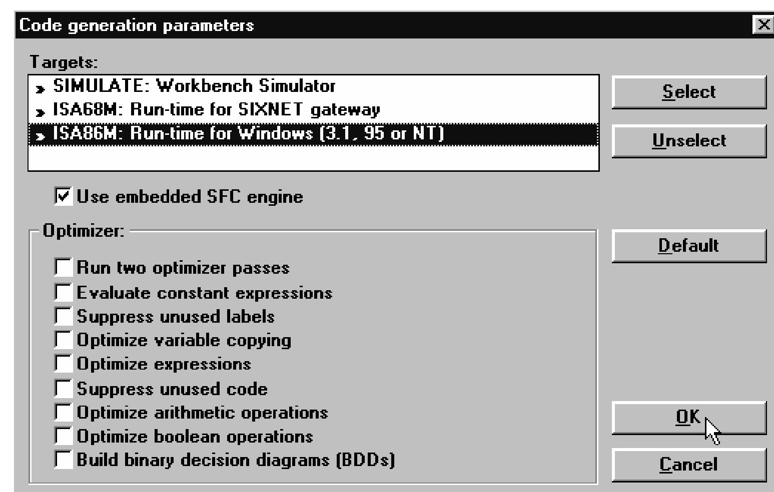
然后 →

选择 **File, Save**

然后 →

选择 **Options, Compiler Options**

确保选中所有三种编译目标。



编译器选项在本手册其他章节有详细的讨论。

现在，确保编译器选项如右图所示，选择 **OK**

然后 →

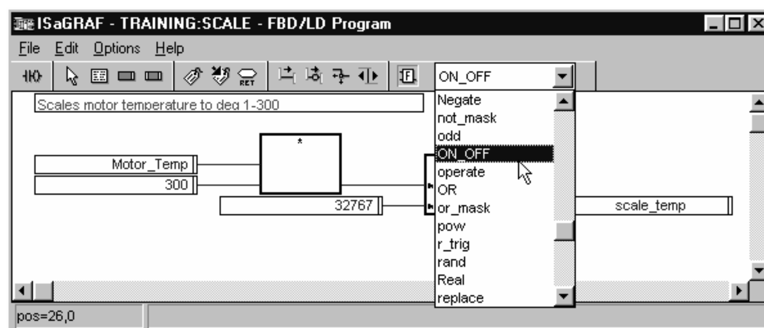
选择 **File, Verify.**

ON_OFF 块现在被编译，并可以在您的 ISaGRAF 项目中使用了。

退出代码生成器。

然后 →

退出程序和 ISaGRAF Libraries工具。



在 **FBD** 编辑器中寻找 **ON_OFF** 功能块：

打开 **Scale** 程序

在 **FBD** 工具条选择下拉框，选中 **ON_OFF**

双击该块，将其放置到程序屏幕任何地方。

选择 **Info** 和 **Notes** ，可看到该块的说明。

选择 **OK** 并不保存退出程序窗口。